Dependency Graphs as a Generic Interface between Parsers and Relation Extraction Rule Learning

Peter Adolphs, Feiyu Xu, Hong Li, and Hans Uszkoreit

DFKI, LT-Lab
Alt-Moabit 91c, 10559 Berlin, Germany
{peter.adolphs,feiyu,lihong,uszkoreit}@dfki.de
http://www.dfki.de/lt/

Abstract. In this paper, we propose to use dependency graphs rather than trees as the interface between a parser and the rule acquisition module of a relation extraction (RE) system. Dependency graphs are much more expressive than trees and can easily be adapted to the output representations of various parsers, in particular those with richer semantics. Our approach is built on top of an existing minimally supervised machine learning system for relation extraction. We extend its original tree-based interface to a graph-based representation. In our experiments, we make use of two different dependency parsers and a deep HPSG parser. As expected, switching to a graph representation for the parsers outputting dependency trees does not have any impact on the RE results. But using the graph-based representation for the extraction with deep HPSG analyses improves both recall and f-score of the RE and enables the system to extract more relation instances of higher arity. Furthermore, we also compare the performance among these parsers with respect to their contribution to the RE task. In general, the robust dependency parsers are good in recall. However, the fine-grained deep syntactic parsing wins when it comes to precision.

1 Introduction

Information extraction (IE) employs various language analysis technologies. The demand for depth of the linguistic analysis is almost parallel to the complexity of the IE task. Shallow components such as part-of-speech tagging and phrase recognition often provide sufficient information for the named entity recognition task. However, for tasks such as relation extraction (RE), the desired setup would be one in which the linguistic parser can provide information about dependencies among linguistic chunks, e.g., grammatical functions or predicate argument structures and the IE system only has to provide corresponding task- or domain-specific interpretation of the linguistic relations and their arguments.
In recent years, parsing systems have achieved great progress with respect to efficiency and robustness (e.g., [4, 8, 10, 23]). In particular, the availability of

large-scale treebanks has given rise to many statistical constituent-based or dependency grammars (e.g., [10]). Meanwhile, the existing hand-written grammars such as Head-driven Phrase Structure Grammars (HPSG) [14] also benefit from statistical parse disambiguation models trained on treebanks for their parse selection and domain adaptation (e.g., [6, 18]). Therefore, more and more IE systems applied large-scale generic linguistic parsers for their RE tasks (e.g., [11, 12]).
An important research area of IE is to develop methods that can learn RE rules automatically from parsing results for a new domain [13]. The aim is to automatically discover rules that map the appropriate phrases in the parsing results to the semantic roles specified in the target relation. In the previous approaches [5, 15–17, 22], the linguistic representations allowed in RE rules are much less expressive than the parsing results. This leads to the loss of many syntactic or semantic structures among the linguistic arguments, which might be essential indications of the mentioning of the target relations.
Our research builds upon an existing system for minimally supervised relation extraction – the DARE system [20, 21]. The DARE system presents a recursive rule representation which supports bottom-up rule learning from dependency parsers for relations with various complexity. The rule learning is embedded in a minimally supervised bootstrapping framework. DARE allows dependency trees with their full expressiveness as interface between the parsers and the rule learning. However, their tree-based representation does not fully support semantic representations containing graph-based predicate argument structures. In this paper, we propose to use dependency graphs as an interface between parsers and the rule learning system, since dependency graphs are expressive enough to be adapted to various syntactic and semantic representations provided by state-of-the-art large-scale parsers. In order to realise this new interface, we extend the DARE system by modifying its rule learning algorithm and its rule representation. In our experiments, we use three parser systems: two dependency parsers – MINIPAR [8] and Stanford Parser [10] – and the HPSG parser PET [1] with a broad-coverage deep linguistic grammar, the English Resource Grammar (ERG) [4]. The two dependency parsers deliver grammatical functions, while the HPSG parser provides graph-based predicate argument structures where an argument can be shared by several predicates. The evaluation results tell us that dependency graphs are very useful to keep as much linguistic information as possible for the rule learning. Without dependency graphs, many linguistic phenomena cannot be covered by the IE systems properly. Thus, it helps to learn more rules, yielding in better recall, in particular, for the HPSG semantic output.
The remainder of the paper is organised as follows: Section 2 discusses related work; Section 3 introduces the DARE system architecture and its rule representation; Section 4 explains the three parsers with their output representations; Section 5 describes the new graph-based DARE rule representation and the corresponding new rule discovery method; In Section 6, various experiments are conducted to compare the dependency graphs as interface with tree-based interface and a systematic quality and error analysis is presented too; Section 7 closes off with a conclusion and discusses the future directions.

2 Related Work

Many minimally supervised or unsupervised approaches targeted to learning rules for extracting complex relations develop their rule representations on top of dependency trees [5, 15–17, 22]. All these representations are less expressive than the dependency trees from which they learn their rules. The learned linguistic patterns for the RE rules are mostly restricted to trees with limited depth and branching factor, e.g., single paths [16] or flat subject-verb-object constructions (binary trees with depth one) [22]. Furthermore, they are only verb-centered. Thus, with this strong constraint, they cannot deal with relation mentions that are expressed in complex nominal compounds or nominalization constructions. Moreover, in comparison to DARE [20], patterns acquired by all these models do not specify the mapping between the linguistic arguments and the relation-specific semantic roles. The DARE rule representation allows the full expressiveness of the dependency trees. But, like other approaches, it cannot deal with semantic representations allowing graph-structures.

For the RE tasks in the biomedical domain, [11, 12] give a detailed evaluation of various parsers. They discover that more semantic-oriented parsing representation such as dependency or predicate-argument structures achieve better overall performance than pure syntactic representation, e.g., phrase structures.

3 DARE: Minimally Supervised Rule Learning for Relation Extraction

DARE [20, 21] is a minimally supervised machine learning system for RE on free texts, consisting of two parts: 1) rule learning and 2) relation extraction (RE). Rule learning and RE feed each other in a bootstrapping framework. The bootstrapping starts from so-called "semantic seeds", which is a small set of instances of the target relation. The rules are extracted from sentences annotated with semantic entity types and parsing results, e.g., dependency structures, which match with the seeds. RE applies acquired rules to texts in order to discover more relation instances, which in turn are employed as seed for further iterations. The core system architecture of DARE is depicted in Fig. 1. The entire bootstrapping stops when no new rules or new instances can be detected. Relying entirely on semantic seeds as domain knowledge, DARE can accommodate new relation types and domains with minimal effort.

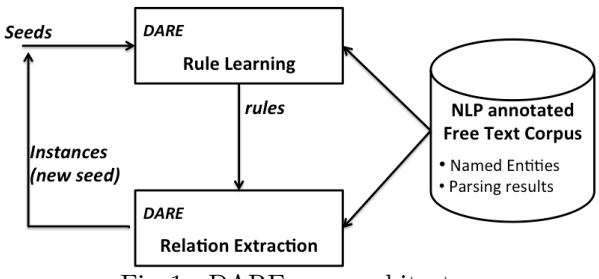


Fig. 1: DARE core architecture

DARE can handle target relations of varying arity through a compositional and recursive rule representation and a bottom-up rule discovery strategy. A DARE rule for an n-ary relation can be composed of rules for its projections, namely, rules that extract a subset of the n arguments. Furthermore, it defines explicitly the semantic roles of linguistic arguments for the target relation. The following examples illustrate the DARE rule and its extraction strategy. *Example 1* is a relation instance of the target relation from [20] concerning Prize awarding event, which contains four arguments: *Winner*, *Prize_Name*, *Prize_Area* and *Year*. *Example 1* refers to an event mentioned in *Example 2.*

Example 1. <*Mohamed ElBaradei, Nobel, Peace, 2005*>.

Example 2. Mohamed ElBaradei, won the *2005 Nobel* Prize for *Peace* on Friday.

Given *Example 1* as a seed, *Example 1* matches with the sentence in *Example 2* and DARE assigns the semantic roles known in the seed to the matched linguistic arguments in *Example 2*. Fig. 2 is a simplified dependency tree of *Example 2* with named entity annotations and corresponding semantic role labelling after the match with the seed. DARE utilises a bottom-up rule discovery strategy to extract rules from such semantic role labelled dependency trees.

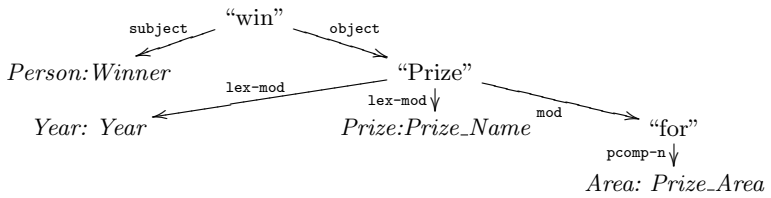


Fig. 2: Dependency tree of *Example 2.* matched with the seed

Rule name :: winner_prize_area_year_1
Rule body ::

$$\left[\begin{array}{ll} \text{head} & \left[\begin{array}{ll} \text{pos} & \text{verb} \\ \text{mode} & \text{active} \\ \text{lex-form} & \text{“win”} \end{array}\right] \\ \text{daughters} & < \left[\text{subject} \quad \left[\text{head} \quad \boxed{1}\textit{Person}\right]\right], \\ & \left[\text{object} \quad \left[\begin{array}{ll} \text{rule} & \text{year\_prize\_area\_1} :: \\ & < \boxed{4}\textit{Year}, \boxed{2}\textit{Prize\_Name}, \\ & \boxed{3}\textit{Prize\_Area} > \end{array}\right]\right] > \end{array}\right]$$

Output :: < $\boxed{1}$*Winner*, $\boxed{2}$*Prize_Name*, $\boxed{3}$*Prize_Area*, $\boxed{4}$*Year* >

Fig. 3: DARE extraction rule.

From the tree in Fig. 2., DARE learns three rules in a bottom-up manner, each step with a one tree depth. The first rule is extracted from the subtree dominated by the preposition "for", extracting the argument *Prize_Area* (*Area*), while

the second rule makes use of the subtree dominated by the noun "Prize", extracting the arguments *Year* (*Year*) and *Prize_Name* (*Prize*), and calling the first rule for the argument *Prize_Area* (*Area*). The third rule "winner_prize_area_year_1" is depicted in Fig. 3. The value of *Rule body* is extracted from the dependency tree. In "winner_prize_area_year_1", the subject value *Person* fills the semantic role *Winner*. The object value calls internally the second rule called "year_prize_area_1", which handles the other arguments *Year* (*Year*), *Prize_Name* (*Prize*) and *Prize_Area* (*Area*).

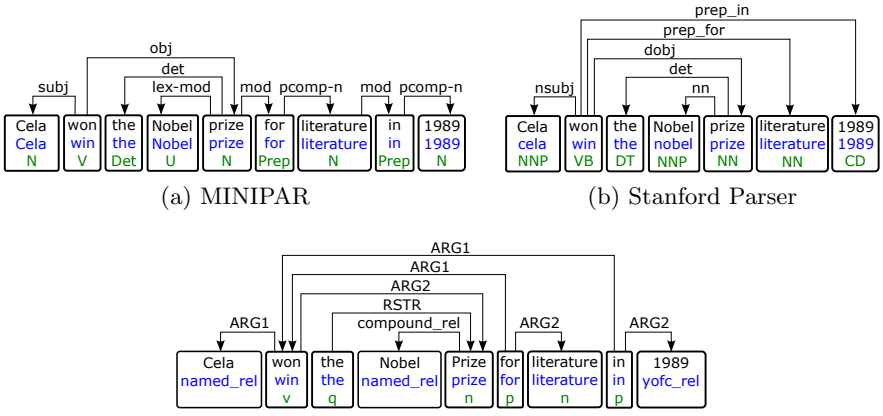
4 Parsers

MINIPAR [8] is a broad-coverage parser for English, implementing a constraint-based parsing algorithm which is reminiscent of chart parsing with rewrite rules[1]. Parse results are available in a dependency tree format. They can also be partial; in this case, the analysis for the sentence itself is a parse forest, consisting of several unconnected dependency trees.

One of the parsing options of the **Stanford Parser**[2] is an unlexicalised PCFG [7], which outputs phrase-structure analyses. These can be converted to labelled dependency representations [9], that are more useful for applications such as IE. Dependency labels denote grammatical functions. The conversion tool can further simplify the dependency structures by collapsing function words, most prominently prepositions, and their associated dependencies, yielding simpler graph structures where content words are directly related to each other via more specialised relation types. Furthermore, dependencies of the head of a conjunction can be optionally propagated to all coordinated elements. Both the collapsing and propagation of dependencies may lead to cyclic structures, i.e. to general dependency graphs.

The **English Resource Grammar (ERG)** is a broad-coverage grammar for English [4], written in the framework of Head-Driven Phrase Structure Grammar (HPSG) [14]. We use the ERG in combination with the efficient HPSG parser PET [1]. Both the ERG and PET are available as Open Source components[3]. Semantics utilised in the ERG is expressed in the Minimal Recursion Semantics (MRS) formalism [3], which essentially encodes a predicate-argument structure with generalised quantifiers and underspecified scopes. MRS representations can be converted to 'Dependency MRS' (DMRS; [2]), and vice versa, a simplified representation that resembles classical dependency structures. In order to gain classical token-to-token dependencies, we further simplify these representations by converting nodes representing compounds to edges and merging overlapping nodes which decompose the meaning of a particular word. The ERG analyses certain constructions systematically differently than the other two parsers. For instance, modification is modeled as modifiers selecting for the heads they modify. Thus, heads with multiple modifiers will have multiple parents. The MRS

[1] `http://webdocs.cs.ualberta.ca/~lindek/minipar.htm`, accessed 26 April 2011
[2] `http://nlp.stanford.edu/software/lex-parser.shtml`, accessed 26 April 2011
[3] `http://www.delph-in.net/`, 26 April 2011



(a) MINIPAR (b) Stanford Parser

(c) ERG DMRS

Fig. 4: Example analyses for the 3 parsers

also provides linguistically more adequate representations of phenomena such as control structures, where phrases are arguments of different predicates at the same time and thus the corresponding nodes have multiple parents. For instance, in the sentence "*John promises us to come*", "*John*" is the subject of "*promise*" as well as the subject of "*come*". In all these cases, the resulting dependency representation will therefore usually be a genuine graph rather than a tree.
Fig. 4 illustrates some of the systematic differences between the parsers. The prepositions in the Stanford analysis are incorporated into a dependency edge and the corresponding node has been eliminated since edges have been collapsed. The reversed dependency directions for the specification and modification structures in the ERG analysis result in nodes with several incoming edges. The actual analyses are structurally quite similar. The analyses of all parsers pick up the correct nodes as the subject and object of the verb. The only real difference for this particular sentence is how the inherently ambiguous choices of attaching the PPs are resolved: while MINIPAR chooses to attach both PPs low, Stanford Parser and ERG attach the PPs high at the verbal node.

5 Dependency Graph as Interface

The strategy to match tree fragments in the dependency structure is inappropriate in cases where relation argument nodes are connected by paths with reversed directions as in Fig. 4 (c). Though it is possible to identify two tree fragments in the structure connecting either the winner, prize and area combination or the winner, prize and year arguments (the trees being rooted at either the preposition "for" or "in", respectively), there is no tree fragment connecting all four arguments. Rather than learning tree rules from dependency trees bottom up, we therefore extend the DARE rule learning algorithm to learn graph rules which identify subgraphs in arbitrary graphs. Fig. 5 shows such a subgraph within the original analysis, connecting all relation arguments (which are highlighted).

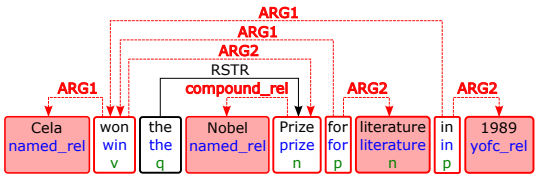


Fig. 5: Graph fragment in an ERG analysis

Since graph-based models provide a general framework for representing all kinds of linguistic information, this strategy also promises to facilitate the combination of different parsing methods using a uniform rule representation.

A DARE graph rule has three components:

1. `rule name`: r_i
2. `output`: a set A containing n arguments of the n-ary relation, labelled with their argument roles.
3. `rule body`: a graph $G = (N, E)$ where N is a set of nodes with – possibly underspecified – features to be matched, and E is a set of – possibly labelled – edges connecting these nodes. The elements of A are coindexed with the reference feature of the corresponding argument nodes in N.

As before, the rule learning happens in two steps. Matching subgraphs are first extracted and then generalised to form extraction rules by underspecifying the nodes and introducing place-holders labelled with the role for the argument nodes. The pattern subgraphs are extracted from the dependency graph by the following procedure:

1. For a given n-ary seed $S = (s_1, \ldots, s_n)$ and a given dependency graph G, collect the set T of all terminal nodes from G that are instantiated with seed arguments in S.
2. For each acceptable combination of seed argument terminal nodes $C = \{t_1, \ldots, t_m\}$ $(m \geq 2)$, find a shortest path S_i between t_i and t_{i+1} for $0 < i < m$.
3. For each combination of seed argument terminal nodes C and the corresponding set of shortest paths $S_C = \{S_1, \ldots, S_m\}$, extract the corresponding pattern subgraph P_C from G, where the set of nodes is the union of the nodes of S_i and the set of edges is the union of the edges of S_i $(0 < i < m)$.

Note that we iterate over all acceptable combinations of argument nodes, where an acceptable combination is one that contains at least two arguments. Further constraints on argument combinations (required arguments) might be desired in order to exclude the extraction of uninformative bits of information. By iterating over all acceptable combinations, we ensure to learn also all projections of the RE rule. Although we do not define an interface for calling RE rules within other RE rules as in the original DARE rule format, this does not affect the performance as it leads to smaller rule set descriptions but not to an increased recognition capacity if no further rule generalisation means are used.

6 Experiments and Evaluation

6.1 Data and Experiment Setup

For several reasons we decided to adopt for our experiments the freely available Nobel Prize award corpus of [21]. The target relation is the 4-ary prize-winning relation <*Winner*, *Prize_Name*, *Prize_Area* and *Year*>. Previous results have shown i) that not every data collection is suited for the minimally supervised approach to RE [20] and ii) that the freely available Nobel Prize award corpus actually has the required properties [19]. The corpus contains 2,864 free text documents from BBC, CNN and NYT, together 143,289 sentences. The total corpus has also a version for evaluation, manually annotated with prize-winning event mentions.

The corpus was automatically preprocessed with named-entity recognition and coreference resolution. Only those sentences are considered for the experiments that can potentially satisfy the following criterion, i.e. that contain at least a person reference and a prize mentioning. The resulting corpus comprises 2,902 potentially relevant sentences for the target relation.

We applied the three parsers described in section 4, namely, MINIPAR 0.5, Stanford Parser 1.6.5, and ERG 1010. The ERG was configured to use the vanilla reading selection model (Redwoods) and a maximum of 1 GB main memory for each sentence. We gained analyses for 2,896 sentences each for MINIPAR and Stanford Parser (99.79% parse coverage), and 2,081 sentences for ERG 1010 (71.71%). The parse coverage for the ERG is lower than the expected $80-90\%$, which is usually observed for texts of similar origins. All parser results were stored in the same dependency graph format.

We performed rule learning and RE on separate subcorpora. Applying the rules to unseen data allows us to judge the quality and reusability of the learned rules. To this end, we split the corpus into two equal-sized parts – the *learning* and the *extraction* corpus. For each parser, we started the bootstrapping process with the same seeds on the learning corpus to learn RE rules. In a second step, we applied the learned rules to each sentence in the extraction corpus to extract relation instances. In a third step, compatible relation instances that are learned from the same sentence are merged, leading to the most specific relation instances for the sentence, that is, only relation instances of higher arity are considered in the evaluation.

Many factors may shape the relation extraction quality of RE rules learned in a boostrapping framework. The domain and data properties and the selection of semantic seeds play an important role of the overall performance of the RE system [19]. In order to eliminate the possibility that the evaluation results are only due to a luckily picked semantic seed, we conducted experiments with different seeds: a) exactly one semantic seed (<*Ahmed Zewail, Nobel, Chemistry, 1999*>), b) 99 randomly chosen semantic seeds, c) all Nobel prize winning events that happened so far. Using all seeds for rule learning is an interesting endevour as it allows us to estimate an upper bound for the RE quality that can be achieved with the current preprocessing tools and learning approach.

Nr. Seeds	*Parser*	*(a) Tree Rules*				*(b) Graph Rules*			
		Prec	Recall	f_1	ØA	Prec	Recall	f_1	ØA
1	MINIPAR	81.97%	46.93%	59.69%	2.77	82.01%	46.69%	59.51%	2.77
	Stanford	79.42%	53.78%	64.13%	2.83	79.48%	53.78%	64.16%	2.84
	ERG	84.06%	34.02%	48.44%	2.80	83.08%	35.22%	49.47%	2.85
99	MINIPAR	81.97%	46.93%	59.69%	2.81	82.01%	46.69%	59.51%	2.81
	Stanford	79.42%	53.78%	64.13%	2.84	79.48%	53.78%	64.16%	2.84
	ERG	84.09%	34.10%	48.53%	2.82	82.99%	35.38%	49.61%	2.86
all	MINIPAR	82.18%	49.00%	61.40%	2.84	82.31%	48.69%	61.18%	2.84
	Stanford	79.58%	54.26%	64.53%	2.88	79.67%	54.26%	64.56%	2.88
	ERG	83.08%	34.74%	48.99%	2.83	82.94%	36.33%	50.53%	2.87

Table 1: Results for the full learning / extraction corpora ($\emptyset A$: average arity)

An event mention extracted from a sentence is considered to be recognised successfully if it is compatible with one of the annotated event mentions available for this sentence. We use standard precision, recall and f_1-score measures for evaluation. In order to assess one of the strengths of the DARE approach, namely its ability to extract relation instances of higher arity, we also calculate the average arity of extracted relations.

6.2 Evaluation

Table 1 shows an evaluation of the RE results of the systems on the full extraction corpus, using (a) tree rules and (b) graph rules with the three different seed sets. As expected, switching from the tree-based to the graph-based relation extractor has no substantial impact on the RE performance using MINIPAR and the Stanford Parser. However, it increases both recall and f-score of RE with the ERG. This is also reflected in the average arity of the extracted instances. While the average arity is virtually unchanged for MINIPAR and Stanford Parser, using graph rules with the ERG helps the system to extract more relation instances of higher arity. It means that graph rules are useful for rich semantic representations and can extract more information than tree rules.
Furthermore, the results confirm the observations made in earlier studies where the same RE task is carried out on these data, namely that the choice of seeds does not substantially influence the RE results for this corpus. Even using one semantic seed can be enough to learn all relevant RE rules. RE with the ERG using the full corpora performs worse than with MINIPAR or the Stanford Parser. This is not surprising since coverage of the ERG on the corpus is much lower than for the other parsers. In order to compare the RE results obtained with the HPSG grammar to the results with the two other parsers more closely, we ran a second series of experiments for all parsers on the HPSG-parsable learning and extraction subcorpora only. In this scenario, all parsers see exactly the same sentences during learning and extraction, levelling the differences due to different paths during bootstrapping. Table 2 shows the results of these experiments with the graph extractor in column (b). Obviously parse coverage is not the only reason for the performance differences in table 1. Though RE with the

Nr. Seeds	*Parser*	*(b) Graph Rules*				*(c) G. R. + Coord. Extr.*			
		Prec	Recall	f_1	ØA	Prec	Recall	f_1	ØA
1	MINIPAR	82.36%	46.54%	59.48%	2.79	81.95%	50.27%	62.32%	2.82
	Stanford	80.26%	53.57%	64.25%	2.86	80.59%	55.87%	65.99%	2.90
	ERG	83.08%	48.52%	61.26%	2.85	81.58%	51.48%	63.13%	2.90
99	MINIPAR	82.36%	46.54%	59.48%	2.83	81.95%	50.27%	62.32%	2.86
	Stanford	80.26%	53.57%	64.25%	2.87	80.72%	55.87%	66.04%	2.90
	ERG	82.99%	48.74%	61.41%	2.86	81.87%	51.48%	63.21%	2.92
all	MINIPAR	82.44%	48.74%	61.26%	2.87	81.72%	51.37%	63.09%	2.89
	Stanford	80.48%	54.88%	65.26%	2.93	80.58%	57.19%	66.90%	2.92
	ERG	82.94%	50.05%	62.43%	2.87	81.85%	52.80%	64.19%	2.91

Table 2: Results for the HPSG-parsable learning / extraction corpora

ERG achieves the best precision scores, using Stanford Parser analyses leads to considerably better recall, which counterweights the lower precision.

A detailed comparison between the RE results obtained with the Stanford Parser and the ERG quickly shows that some of the mismatches were due to systematically different coordination analyses. While MINIPAR and the Stanford Parser anchor an incoming dependency to a coordinated NP at the first conjunct and then link the remaining conjuncts with a special conjunction dependencies, the ERG creates explicit (for words such as "and") or implicit (covert) conjunction nodes which link the conjuncts and places the incoming dependency at the head conjunction node. Given an RE rule learned from a structure without coordination, this analysis allows MINIPAR and the Stanford Parser to extract the first conjunct in a similar structure with coordination, while the ERG cannot extract anything at the target node with a corresponding rule as the coordination node will not match the argument node in the rule. We therefore further extended the graph extractor to interpret coordination structures during extraction. During rule matching, the extractor may follow any coordination links found in the graph. This strategy is applied for all parsers, and column (c) in table 2 shows that all parsers benefit from this extended extraction strategy. Although this extraction strategy has been a useful step to improve the RE results, it is also apparent that the differences in the coordination analysis are not responsible for the performance differences between the parsers. Given enough learning data, rules for extracting from coordinated structures will be learned with the DARE learning approach.

The remaining possible reasons for the differences between parsers on the same corpus are different grammar coverage, i.e. missing analyses for certain constructions, different strengths of the reading selection models and suitability of the granularity of analysis for the RE task. Since we do not have gold treebanks for the three parsers yet, we cannot systematically assess grammar coverage and reading selection quality. However, we saw a tendency during our qualitative comparison of the RE results with the Stanford Parser and the ERG that the Stanford dependency representation, which provides semantically motivated interpretations and also flatter analyses for some linguistic structures, is beneficial for the RE task.

7 Conclusion and Future Work

In order to faithfully express the semantics of linguistic phenomena such as multiple modifiers of a head word or raising and control constructions, graph structures are needed for representation. We have extended the interface between DARE and parsers from trees to generic dependency graphs, allowing us to deal with the output representations of various parsers, in particular those with rich semantics. Our experiments confirm that the graph-based interface is expressive enough for learning extraction rules exhaustively from linguistic analyses provided by various parsers. As expected, switching to a graph representation for the dependency tree parsers does not have any impact on the RE results. But using the graph-based representation for the extraction with deep HPSG analyses improves both recall and f-score of the RE and the arity of the extracted instances is higher, i.e., more information can be detected. During our experiments, we also discover that the Stanford dependency representation is well designed for semantically oriented NLP applications. Last but not least, we have demonstrated the general applicability of a mature deep grammar for English, the ERG, in such a task. It is impressive to see the big steps forward towards real-world applications that the grammar made over the last years.

For the future, we plan an extensive empirical analysis of the parser differences w.r.t. success or failure in the RE task. If we succeed to pin down the advantages of each parser to distinguishing criteria, we can improve the quality of RE by learning RE rules from the merged output of a parser ensemble. Finally, we plan to exploit the rich modelling of important but challenging semantic relations in the ERG such as modality, negation and their scopal interactions by directly operating on the MRS representations.

Acknowledgements This research was conducted in the context of the DFG Cluster of Excellence on Multimodal Computing and Interaction (M2CI), project KomParse (funded by the ProFIT program of the Federal State of Berlin and the EFRE program of the EU, contract 1014 0149), projects Theseus Alexandria and Alexandria for Media (funded by the German Federal Ministry of Economy and Technology, contract 01 MQ 07 016), and project TAKE (funded by the German Federal Ministry of Education and Research, contract 01IW08003).

References

1. Callmeier, U.: Preprocessing and encoding techniques in PET. In: Oepen, S., Flickinger, D., Tsujii, J., Uszkoreit, H. (eds.) Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing. CSLI Publications (2002)
2. Copestake, A.: Dependency and (R)MRS (Dec 2008), `http://www.cl.cam.ac.uk/~aac10/papers/dmrs.pdf`, unpublished Draft. December 9, 2008
3. Copestake, A., Flickinger, D., Pollard, C., Sag, I.A.: Minimal recursion semantics: An introduction. Research on Language and Computation 3(4) (2005)
4. Flickinger, D.: On building a more efficient grammar by exploiting types. Natural Language Engineering 6(1) (2000)

5. Greenwood, M., Stevenson, M., Guo, Y., Harkema, H., Roberts, A.: Automatically Acquiring a Linguistically Motivated Genic Interaction Extraction System. Proc. of the 4th Learning Language in Logic Workshop (LLL05), Bonn, Germany (2005)
6. Hara, T., Miyao, Y., Tsujii, J.: Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In: Second International Joint Conference on Natural Language Processing (2005)
7. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Advances in Neural Information Processing Systems 15 (NIPS 2002), vol. 15. MIT Press, Cambridge, MA (2003)
8. Lin, D.: Dependency-based evaluation of MINIPAR. In: Abeillé, A. (ed.) Treebanks - Building and Using Parsed Corpora. Kluwer Academic Publishers (2003)
9. de Marneffe, M., Maccartney, B., Manning, C.: Generating typed dependency parses from phrase structure parses. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006). Genoa, Italy (2006)
10. de Marneffe, M., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation. Manchester, UK (2008)
11. Miwa, M., Pyysalo, S., Hara, T., Tsujii, J.: A comparative study of syntactic parsers for event extraction. In: Proceedings of the 2010 Workshop on Biomedical Natural Language Processing. Uppsala, Sweden (July 2010)
12. Miyao, Y., Sagae, K., Satre, R., Matsuzaki, T., Tsujii, J.: Evaluating contributions of natural language parsers to protein-protein interaction extraction. In: Bioinformatics. vol. 25 (2009)
13. Muslea, I.: Extraction patterns for information extraction tasks: A survey. In: AAAI Workshop on Machine Learning for Information Extraction. Orlando, Florida (July 1999)
14. Pollard, C.J., Sag, I.A.: Head-Driven Phrase Structure Grammar. University of Chicago Press (1994)
15. Stevenson, M., Greenwood, M.A.: Comparing information extraction pattern models. In: Proceedings of the Workshop on Information Extraction Beyond The Document. Association for Computational Linguistics, Sydney, Australia (July 2006)
16. Sudo, K., Sekine, S., Grishman, R.: An improved extraction pattern representation model for automatic IE pattern acquisition. Proceedings of ACL 2003 (2003)
17. Sudo, K., Sekine, S., Grishman, R.: Automatic pattern acquisition for japanese information extraction. In: Proceedings of the First International Conference on Human Language Technology Research (HLT '01). Morristown, NJ, USA (2001)
18. Toutanova, K., Manning, C.D., Flickinger, D., Oepen, S.: Stochastic HPSG parse selection using the Redwoods corpus. Journal of Research on Language and Computation 3(1) (2005)
19. Uszkoreit, H., Xu, F., Li, H.: Analysis and improvement of minimally supervised machine learning for relation extraction. In: 14th International Conference on Applications of Natural Language to Information Systems. Springer (2009)
20. Xu, F.: Bootstrapping Relation Extraction from Semantic Seeds. Phd-thesis, Saarland University (2007)
21. Xu, F., Uszkoreit, H., Li, H.: A seed-driven bottom-up machine learning framework for extracting relations of various complexity. In: Proceedings of ACL 2007. Prague, Czech Republic (2007)
22. Yangarber, R.: Scenario Customization for Information Extraction. Dissertation, New York University, New York, USA (2001)
23. Zhang, Y.: Robust Deep Linguistic Processing. Phd-thesis, Saarland University, Saarbruecken, Germany (2007)