Local ambiguity packing and discontinuity in German

Berthold Crysmann
DFKI GmbH & Saarland University
Stuhlsatzenhausweg 3
D-66123 Saarbrücken
crysmann@dfki.de

Abstract

We report on recent advances in HPSG parsing of German with local ambiguity packing (Oepen and Carroll, 2000), achieving a speed-up factor of 2 on a balanced test-suite. In contrast to earlier studies carried out for English using the same packing algorithm, we show that restricting semantic features only is insufficient for achieving acceptable runtime performance with a German HPSG grammar. In a series of experiments relating to the three different types of discontinuities in German (head movement, extraction, extraposition), we examine the effects of restrictor choice, ultimately showing that extraction and head movement require partial restriction of the respective features encoding the dependency, whereas full restriction gives best results for extraposition.

1 Introduction

It is a well-known fact that chart parsing without techniques for local ambiguity packing (Earley, 1970) faces a combinatorial explosion of the search space, owing to the (structural) ambiguity imminent to natural language. Thus, identical edges with different internal derivation history can be packed into a single representative for further processing, thereby effectively solving the complexity issue. In context-free grammars augmented with a unification formalism, packing based on the CF symbol equality has been complemented by subsumption- or disjunction-based packing of the associated feature structures (Moore and Alshawi, 1992; Maxwell and Kaplan, 1995). For parsing with constraint-based grammars, such as HPSG, which do not possess an explicit context-free backbone, (Oepen and Carroll, 2000) have proposed an efficient packing algorithm based on feature structure subsumption only.

In contrast to the symbols in context-free grammars, feature structures in unification-based grammars often include information encoding (part of) the derivation history, most notably semantics. In order to achieve successful packing rates, feature restriction (Shieber, 1985) is used to remove this information during creation of the packed parse forest. During the unpacking phase, which operates only on successful parse trees, these features are unified back in again.

For their experiments with efficient subsumption-based packing, (Oepen and Carroll, 2000) experimented with different settings of the packing restrictor for the English Resource Grammar ERG (Copestake and Flickinger, 2000): they found that good packing rates, and overall good performance during forest creation and unpacking were achieved, for the ERG, with partial restriction of the semantics, e.g. keeping index features unrestricted, since they have an impact on external combinatorial potential, but restricting most of the internal MRS representation, including the list of elementary predications and scope constraints. Restriction of syntactically potent features, has thus been found both unnecessary and less efficient.

First experiments in ambiguity packing with a German HPSG grammar (GG; http://gg.dfki.de) revealed that restriction of semantics only does not give rise to any acceptible results in terms of runtime performance. It became clear quite quickly that the

Proceedings of the 5th Workshop on Important Unresolved Matters, pages 144–151,
Ann Arbor, June 2005.

bulk of failing subsumptions impeding creation of a sufficiently compact forest were related to two syntactic features, SLASH and DSL. In German, these features contain references to non-empty valence lists, which eventually wind up encoding derivation history. Using a more aggressive restrictor to eliminate these features during forest creation did not show the desired performance either: owing to massive overgeneration, the resulting forest was either not compact enough, or most of the efficiency gains were wasted on unpacking failures in the second phase.

In this paper we report on recent advances with local ambiguity packing for German, showing how partial restriction can achieve good packing rates at negligible unpacking cost, yielding an overall speedup by a factor of 2, as compared to parsing without ambiguity packing. Running a series of experiments with different restrictor setting for three different features involved with non-local dependencies we examine in detail how the choice of restrictor affects the observable performance. The paper is organised as follows: section 2 will give an overview of the relevant syntactic constructions of German, and their implementation in GG. Section 3 gives a description of the experimental setup (3.1), followed by a discussion of the main results (3.2), detailing how different settings for feature restriction affect parsing performance.

2 Discontinuity in German

Head movement German, in contrast to English is a verb-final language with a verb-second effect, that is, non-finite verbs are standardly placed sentence-finally. In clauses other than complementizer-introduced subclauses and relative clauses, the finite verb surfaces in a clause-initial position (either first or second). Any major constituent may occupy the topic position preceding the finite verb, including subject, complements or modifiers.

Owing to the V2 effect, the parts of a verb cluster are discontinuous. Since both the finite verb and the non-finite verb cluster impose constraints on constituents in the Mittelfeld, standard approaches to German syntax in HPSG assume, since (Kiss and Wesche, 1991), that the initial verb is related to the final verb cluster by means of head movement: clause-finally, a trace is inserted that combines the

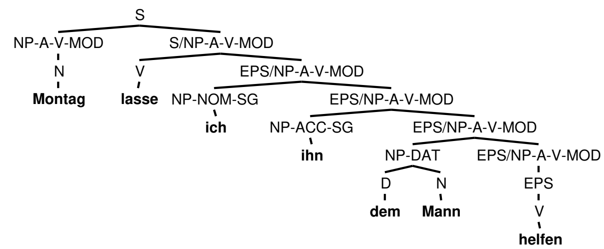

Figure 1: DSL: *Monday let he him the man help*

argument structure of the final cluster with the subcategorisation requirements percolated down from the finite verb using a special feature DSL (="double SLASH"). Arguments in the Mittelfeld are saturated as complements of the clause-final trace. The grammar used here assumes head movement for discontinuous predicates (Crysmann, 2003), following in this respect the earlier implementation by (Müller and Kasper, 2000). In order to relate the initial verb to the verb cluster and its arguments in the Mittelfeld, like the subject and direct object in figure 2, the DSL (or V1) feature must percolate subcategorisation requirements for subject and object, as well as for the verb cluster. At the gap site, the valence information percolated via DSL is inserted into the actual valence lists of the verb trace. Since the requirements are matched against actual arguments found in the Mittelfeld, the valence lists contained in DSL get instantiated to whatever argument it satisfies, thereby creating a partial representation of derivation history. While theoretically, this is just the right behaviour, it has clear repercussions for parsing with ambiguity packing.

Partial VP fronting Another aspect, in which the syntax of German differs from that of English is in the area of extraction: while in English only constituents with a saturated COMPS list can undergo wh-movement, this is not the case in German: as shown in figure 2, the verb `schenken` 'give/donate' has been fronted, leaving its two complements behind.

In HPSG, partial VP fronting is analysed by a combination of two mechanisms (Müller, 1999; Nerbonne, 1994): first, standard argument composition in the verb cluster, following (Hinrichs and Nakazawa, 1990), combined with a standard SLASH-based treatment of long-distance extraction.

Again, since argument composition is performed

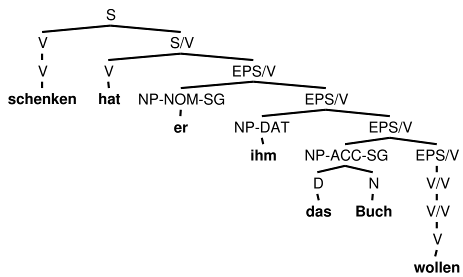

Figure 2: SLASH: *give has he him the book wanted*

by strcuture-sharing, i.e. reentrancy between the valence list of the governing predicate and the unsaturated valence list of the governed predicate, extraction of the governed predicate by means of SLASH percolation carries this reentrancy over into SLASH. From a general linguistic point of view, this is highly desirable, because valence requirements of the extracted verb must be matched against the arguments that satisfy them in the Mittelfeld. The only drawback is, that we are confronted, again, with a syntactic feature containing, among other things, records of derivation history.

3 Evaluation

3.1 Test setup

In order to systematically investigate the effect of restriction of syntactically potent features on the parsing efficiency with local ambiguity packing, we created a test field consisting of 8 different parameter settings (out of 27 logically possible settings): 1 run without packing, 1 run with optimal settings for the three features under consideration, and 2 runs with suboptimal settings for each of the three features.

All test runs were performed on a balanced test suite extracted from the Verbmobil corpus, using 100 items per input length, from sentence length 1 to 22, thus totalling 2200 test items. Although the Verbmobil corpus does contain test sentences of up to 70 words long, their number drops quite quickly from sentence length 23 on.

The parser used in the experiments is the current SVN version of Pet (Callmeier, 2000), running the March 24 version of GG (http://gg.dfki.de; (Müller and Kasper, 2000; Crysmann, 2003; Crysmann, 2005)). Tests were run on an Intel Core Duo machine using a single T2600 CPU at 2.16GHz with 2 GB main memory.

To ensure that we can study parser performance on input of increasing length, we used a rather generous upper limit of 150,000 passive edges. Taking as a guideline the average space consumption per edge of the non-packing parser, we calculated that parsing could still be done comfortably in main memory, i.e., without using swap space.

All measurements were performed using the [incr tsdb()] profiling platform (Oepen and Flickinger, 1998). Parsing times reported are total CPU times (in seconds), including *exhaustive* unpacking of the parse forest, whenever applicable.

3.2 Results

The main result of our study is that local ambiguity packing in constraint-based parsing of German can lead to performance improvements, once feature restriction is extended from purely semantic features to syntactically potent features used to model discontinuity, such as SLASH, DSL, and ANC (see below). We also found that positive performance effects could only be achieved, if SLASH and DSL features were partially restricted in such a way as to only eliminate all records of derivation history (in terms of instatiated subcategorisation lists), while retaining most of the other constraints represented in these features.

Compared to a non-packing baseline parser with feature structure unfilling, we observed an overall speed-up by a factor of 2 with local ambiguity packing on a balanced test suite. As shown by figure 3.2, local ambiguity packing with optimal restrictor settings is effective in taming the combinatorial explosion of the search observed by the non-packing parser.

Analogous to the reduction in search space, performance savings grow continuously with increasing input length: from sentence length 14 onwards (factor 0.84) relative processing time decreases continually up to a factor of 0.27 at sentence length 22. With an average CPU time of 0.69s at sentence length 22, performance is by far better than real-time behaviour. Note further, that the non-packing parser benefits here from a ceiling effect: with 25 out of 2200 test items (1%), the available resources of 150,000 passive chart edges were exhausted before the search space was fully explored. With ambiguity packing under an appropriate restrictor, by contrast, the search space was fully explored.

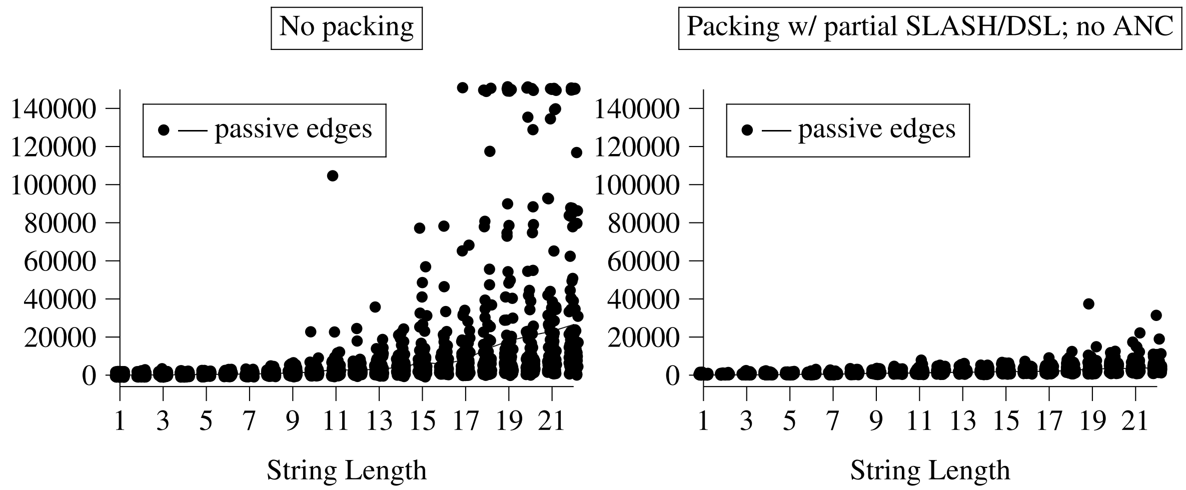

Figure 3: Comparison of chart size relative to input length

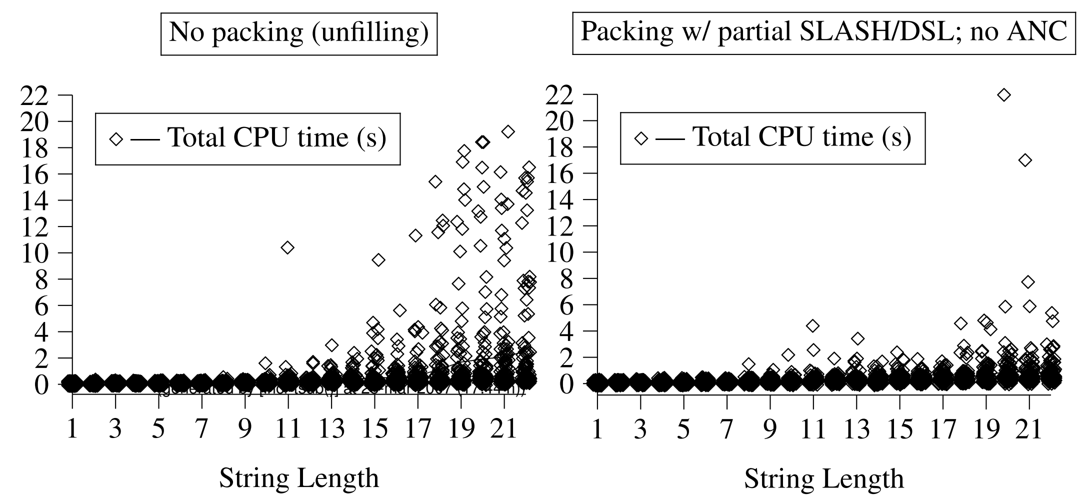

Figure 4: Comparison of processing time relative to input length

Restricting DSL The first syntactically potent feature investigated in these experiments is the feature DSL (or V1), which serves to relate, by means of simulated head movement, the finite verb in clause-second position to the clause-final verb cluster. Essentially, this feature is used to pass down the valence information from the initial verb to the clause-final verb trace, where this valence information is combined with that of the cluster. In the grammar under consideration, verb movement is restricted to discontinuous verb clusters (Crysmann, 2003), i.e., to situations where there is either an overt verb cluster, or a stranded verb particle in the right sentence bracket.

Since actual or putative arguments of the verb trace must be checked against the actual valence information of the V2 verb, derivation history must be carried along as part of the DSL feature.

Obviously, any feature that (partially) encodes derivation history is a potential threat to efficient ambiguity packing. We therefore experimented with three different settings regarding restriction of this feature: full restriction, no restriction, and a partial restriction, where only constraints pertaining to HEAD information of the final cluster were retained, such as category, or form (most crucial for stranded particles).

Results are summarised in table 1. Besides the feature studied here, the restrictor includes the se-

[1]Here, and in the following two tables $\equiv$ stands for packing under equivalence, $\sqsupset$ for proactive packing, $\sqsubset$ for retroactive packing, and $\perp$ for freezing.

	Edges	Time (s)	Unpack (s)	Subsumption	$\equiv$	$\sqsupset$	$\sqsubset$	$\perp$	Factor (time)	Subs. cost	Pack rate
Unfill	6424	0.56	0	0	0	0	0	0	1	N/A	0
Partial DSL	1494	0.28	0.01	36404.15	307.28	193.33	36.67	335.84	**0.5**	67.76	0.36
Full DSL	1832	1.96	0.01	363840.47	186.19	111.31	42.96	251.32	3.5	1068.68	0.19
No DSL	1917	0.61	0.01	106392.57	568.34	484.68	80.8	926.79	1.09	93.83	0.59

Table 1: Performance of packed parsing with different restriction of DSL[1]

mantic features like RELS and HCONS, as in (Oepen and Carroll, 2000), as well as optimal settings for SLASH and ANC.

Leaving DSL unrestricted features the lowest number of packings, amongst the three settings, both in absolute terms, and in relative packings per edge (0.19). As a consequence, average chart size is bigger than with either partially or fully restricted DSL. Another negative behaviour of packed parsing with an unrestricted DSL is the incommensurate number of subsumption tests carried out: at a ratio of 1068.68 subsumption tests per packing, this accounts chiefly for the inefficiency, in particular, when compared to the much more moderate rates of 67.76 and 93.83 achieved with partially restricted and fully restricted DSL. Thus, even though overall chart size is reduced when compared to parsing without ambiguity packing, these savings in space are not sufficient enough to pay off the overhead incurred by testing for subsumption. As a net effect, overall parsing time is 3.5 times longer compared to the non-packing baseline.[2]

Fully restricting DSL by contrast yields a very good packing rate (0.59) at moderate costs in terms of subsumption test per packing (93.83). However, with the grammar not being restrictive enough during forest creation, overall chart size is bigger (1832 passive edges) than with partially restricted DSL (1494). Best results are obtained with partially restricted DSL, where derivation history in terms of actual or putative arguments of the verb trace is removed, but reference to HEAD information of the final cluster is maintained, thereby ensuring that the initial verb only combines with appropriate verb clusters. This not only leads to the most compact chart, but also features the lowest number of subsumption tests, both absolute and relative. In sum, partial restriction of DSL was the only setting that actually beat the baseline defined by parsing without ambiguity packing.

[2]Edges in packed parsing are actually more costly than in parsing without ambiguity packing. Since efficient subsumption checking and feature structure unfilling are mutually exclusive, edges in general consume much more space when parsing with ambiguity packing, increasing the cost of copying in unification.

Restricting SLASH The second experiment we carried out relates to the feature SLASH, used for long-distance dependencies. Owing to the V2 effect in German, constituents in the clause-initial preverbal position are typically placed there by means of extraction, including unmarked subjects. This differs quite clearly from English, where standard SVO order does not involve any movement at all. Another striking difference between the two languages is that German, but not English permits fronting of partial VPs: in complex predicates, as witnessed with modals and control verbs, as well as in auxiliary-participle combinations, the downstairs predicate can be fronted, leaving part (or even all) of its complements to be realised in the Mittelfeld. Since German is a non-configurational language, pretty much any combination of fronted vs. stranded complements can be found, in any order. In GG, partial VP fronting is effected by special extraction rules, which removes the valency of pertaing to the fronted verb from the subcategorisation list of the embedding predicate, and inserts it into SLASH. Simultaneously, the remaining complements of the embedding verb are composed with the locally underspecified subcategorisation list of the extracted verbal complement. In order to match the subcategorisation requirement of the extracted verb with those of its complements that are realised in the Mittelfeld, the subcategorisation list must be percolated via SLASH as well. Since elements on the subcategorisation list in SLASH are reentrant with elements on the composed subcategorisation list of the embedding predicate, the former gets specified to the complements that saturate the requirements in the Mittelfeld. As a result, we observe a massive encoding of derivation history in SLASH.

Besides the rules for partial VP fronting, the grammar recognises 3 more extraction rules, one for subject, one for non-subject complements, and one for adjuncts. Out of these three, only adjunct extraction rules encode reference to their extraction context in SLASH: since modifiers select the heads they adjoin to via a feature MOD, which is reentrant with the SYNSEM of that head, they inevitably carry along a good deal of that head's derivation history.

We tested three different configurations of the restrictor: one with unrestricted SLASH, one where the entire SLASH feature was removed during forest creation, and a partially restricted variant. This partially restricted variant preserves the full SLASH representation for ordinary subject and complement extraction, but uses an impoverished representation for adjunct extraction and partial VP fronting. Technically, this was achieved by using two SLASH features in parallel: an auxiliary, impoverished _SLASH to be used during forest creation, and the full SLASH feature during unpacking. For adjunct extraction and partial VP fronting, _SLASH contains type restrictions on the head value of the fronted element, together with restrictions on the saturation of valence lists, if applicable.[3] For subject and complement extraction _SLASH contains the same information as SLASH. In sum, partial restriction tries to maximise restrictiveness in those case, where no reference to the extraction context is encoded in SLASH, while at the same time it minimises encoding of derivation history in the other cases, by replacing token identity with somewhat weaker type constraints.

The results of this second experiment are summarised in table 2. Again, we have used optimal settings for DSL and ANC, as established by independent experiments.

Parallel to our observations regarding the restriction of DSL, we observe that performance is worst for packed pasring with a completely unrestricted SLASH feature: not only is the packing rate quite low (0.25 packings per edge), but also the costs for packing in terms of the number of subsumption checks carried out is highest amongst all the experiments reported on in this paper, peaking at 1355.85 subsumption tests per successful packing. The impact on chart size is slightly worse than what we observed with an unrestricted DSL feature. In sum, the suboptimal packing together with the excessive subsumption costs account for the fact that this setting performs more than 8 times as badly as the baseline.

Although packed parsing with fully restricted SLASH performs much better than having SLASH entirely unrestricted, it still falls short of the baseline by a factor of 1.36. This is due to several reasons: first, although the packing rate is good (0.59), the chart is the biggest observed with packed parsing in all the experiments carried out, being more than 2 times as big as the parse chart with optimal restrictor settings. This is mainly due to the fact that the grammar is far to unconstrained during forest creation, allowing too many inconsistent analyses to enter the chart. This is also corroborated by the fact that this is the only test run where we experienced a noticeable increase in unpacking time. Another observation, for which we cannot offer any explanation at present, pertains to the increased cost associated with retroactive packing: the amount of frezing that has to be done for edges masked by retroactive packing is far higher than any other value found in these experiments.

In a separate test run, we used simultaneous full restriction for DSL and SLASH, in order to verify whether our assumtion that the choice of one restrictor is independent from the others. By and large, our hypothesis was confirmed: having both DSL and SLASH fully restricted performed more than 2.5 times worse than full restrcition of SLASH whith partial restriction of DSL.

Still in parallel to our findings regarding DSL, partial restriction of SLASH performs best, confirming that the compromise between restrictiveness and eleimination of derivation history is effective to achieve a runtime behaviour that clearly outperforms the baseline. The packing rate achieved with partial restriction of semantics, DSL and SLASH (0.36) is actually very close to the packing rates reported in (Oepen and Carroll, 2000) for the ERG, which figures around 0.33 for input longer than 10 words. Also, the compactness of the chart with input of increasing length (cf. figure 3.2), and the low number (2) of performance outliers (cf. figure 3.2) suggest that we are indeed close to optimal feature restriction.

Decisions on which features to preserve within SLASH under partial restriction were mainly de-

[3] E.g., modifiers must have saturated valence lists, whereas fronted partial VP constituents may have open valencies relating to complements in the Mittelfeld.

	Edges	Time (s)	Unpack (s)	Subsumption	$\equiv$	$\sqsupset$	$\sqsubset$	$\perp$	Factor (time)	Subs. cost	Pack rate
Unfill	6424	0.56	0	0	0	0	0	0	1	N/A	0
Partial SLASH	1494	0.28	0.01	36404.15	307.28	193.33	36.67	335.84	**0.5**	67.76	0.36
Full SLASH	2187	4.72	0.01	728385.4	314.66	149.21	73.35	826.1	8.43	1355.85	0.25
No SLASH	3435	0.76	0.16	97965.05	883.79	994.87	145.44	2583.51	1.36	48.4	0.59

Table 2: Performance of packed parsing with different restriction of SLASH

rived in a test-debug cycle. We therefore plan to investigate different configurations of partially restricted SLASH in future work.

Restricting ANC The last experiment we carried out relates to the ANC (=ANCHOR) feature used to percolate semantic attachment anchors for relative clause extraposition in the style of (Kiss, 2005; Crysmann, 2005). Using ANC, index and handle of each and every NP are collected and passed up the tree, to be bound by an extraposed relative clause attached to the same subclause.

Again, we tested three different settings: full restriction of all 3 anchor feature (SELF, ACTIVE, INERT), no restriction, and partial retsriction, where the elements on the lists were restricted to *top*, thereby recording only the number of percolated anchors, but not their nature in terms of index features. ANC features encode derivation history in two ways: first, structurally higher anchors (NPs) are represented at the front of the list, whereas more deeply embedded anchors are found further down the list. Second, to control for spurious attachments, only anchors inherited from a left daughter are accessible for binding (ACTIVE), the others remain on the INERT list. Both the order of indices on the lists, list length and the distribution of anchors over ACTIVE and INERT lists partially encode constituent-internal structure.

Results of this experiment are summarised in table 3.

Similar to our two previous experiments, entirely unrestricted ANC behaves worst, but nowhere nearly as bad as having SLASH or DSL unrestricted. In fact, relative packing rates achieved by all three restrictor settings are by and large the same in this experiment. The main difference between unrestricted ANC concerns the overall compactness of the forest and the number of subsumption test performed.

Partial restriction already performs better than unrestricted ANC: since partially restricted ANC does not record the nature of the anchors, at least one way in which derivation history is recorded is effectively masked.

Contrary to our previous experiments, however, partial restriction does not outperform full restriction. Although this finding comes somewhat at a surprise, there is nevertheless a straightforward explanation for the difference in behaviour: while full restriction necessarily improves chart compactness, the adverse effects of full restriction do not come to bear as often as in the case of fully restricted SLASH or DSL, since attachment of extraposed relative clauses presupposes the existence of an already constructed chart edge for the relative clause. In contrast to extraction and head movement, which can be found in practically every sentence-size test item, distribution of relative clauses is comparatively low. Furthermore, constituents serving as fillers for SLASH or DSL dependencies can actually be quite small in size and different in shape, which increases the potential for overgeneration with fully restricted movement features. Relative clauses, on the other hand, are always clause-sized, and their properties depend on the information percolated in ANC only to a very little degree (namely number and gender agreement of the relative pronoun).

4 Conclusion

In this paper, we have explored the effects in the choice of restrictor for HPSG parsing of German with local ambiguity packing. Based on initial observation that a semantics-only restrictor gives suboptimal runtime performance in packed parsing, we found that three features representing discontinuities were mainly responsible for inefficiency with local ambiguity packing, namely SLASH for extraction, DSL for head movement, and ANC for relative clause extraposition, all of which may encode part of the derivation history.

We have shown that partial restriction of SLASH and DSL features, together with full restriction of ANC yields satisfactory parsing performance

	Edges	Time (s)	Unpack (s)	Subsumption	$\equiv$	$\sqsupset$	$\sqsubset$	$\perp$	Factor (time)	Subs. cost	Pack rate
Unfill	6424	0.56	0	0	0	0	0	0	1	N/A	0
Partial ANC	1586	0.37	0.01	55392.34	319.35	232.28	51.34	608.51	0.66	91.87	0.38
Full ANC	1704	0.58	0.01	104699.81	346.35	257.92	64.66	758.27	1.04	156.52	0.39
No ANC	1494	0.28	0.01	36404.15	307.28	193.33	36.67	335.84	**0.5**	67.76	0.36

Table 3: Performance of packed parsing with different restriction of ANC

with ambiguity packing, outperforming the a non-packing baseline parser with feature structure unfilling by a factor of 2. Even more importantly, combinatorial explosion at increasing input length is effectively tamed, such that performance gains improve with longer input sentences.

Acknowledgement

The research reported on in this paper has been carried out as part of the DFKI project Checkpoint, funded by the Federal State of Berlin and the EFRE programme of the European Union. I am also gratefully indepted to Bernd Kiefer for his support of the runtime parser and his expert advice. Many thanks also to Ulrich Callmeier, Dan Flickinger, Stefan Müller, Geert-Jan van Noord, and Stephan Oepen, for their comments and suggestions.

References

Ulrich Callmeier. 2000. PET — a platform for experimentation with efficient HPSG processing techniques. *Journal of Natural Language Engineering*, 6(1):99–108.

Ann Copestake and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000)*, Athens.

Berthold Crysmann. 2003. On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP 2003*, pages 112–116, Borovets, Bulgaria.

Berthold Crysmann. 2005. Relative clause extraposition in German: An efficient and portable implementation. *Research on Language and Computation*, 3(1):61–82.

J. Earley. 1970. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102.

E. Hinrichs and T. Nakazawa. 1990. Subcategorization and VP structure in German. In Hughes, Shaun, and Salmons, editors, *Proceedings of the Third Symposium on Germanic Linguistics*, Amsterdam. Benjamins.

Tibor Kiss and Birgit Wesche. 1991. Verb order and head movement. In Otthein Herzog and Claus-Rolf Rollinger, editors, *Text Understanding in LILOG*, number 546 in Lecture Notes in Artificial Intelligence, pages 216–240. Springer-Verlag, Berlin.

Tibor Kiss. 2005. Semantic constraints on relative clause extraposition. *Natural Language and Linguistic Theory*, 23:281–334.

John T. Maxwell and Ronald M. Kaplan. 1995. A method for disjunctive constraint satisfaction. In Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell, III, and Annie Zaenen, editors, *Formal Issues in Lexical-Functional Grammar*, pages 381–401, Stanford University. CSLI.

R. C Moore and H. Alshawi. 1992. Syntactic and semantic processing. In H. Alshawi, editor, *The Core Language Engine*, pages 129–148. The MIT Press, Cambridge, MA.

Stefan Müller and Walter Kasper. 2000. HPSG analysis of German. In Wolfgang Wahlster, editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 238–253. Springer, Berlin.

Stefan Müller. 1999. *Deutsche Syntax — deklarativ*. Linguistische Arbeiten. Niemeyer, Tübingen.

John Nerbonne. 1994. Partial verb phrases and spurious ambiguities. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German in Head-Driven Phrase Structure Grammar*, number 46 in Lecture Notes, pages 109–150. CSLI Publications, Stanford University.

Stephan Oepen and John Carroll. 2000. Ambiguity packing in constraint-based parsing - practical results. In *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics*, pages 162–169, Seattle, WA.

Stephan Oepen and Dan Flickinger. 1998. Towards systematic grammar profiling. test suite technology ten years after. *Journal of Computer Speech and Language*, 12:411–436.

Stuart Shieber. 1985. Using restriction to extend parsing algorithms for complex feature-based formalisms. In *Proceedings of 23rd meeting of the Association of Computational Linguistics*, pages 145–152, Chicago, IL.