Towards efficient HPSG generation for German, a non-configurational language

Berthold CRYSMANN[1] *Woodley PACKARD*[2]
(1) LLF (UMR 7110), CNRS & U Paris–Diderot, 5 rue Thomas Mann, F-75205 Paris Cedex 13
(2) University of Washington, Seattle, WA, USA
crysmann@linguist.jussieu.fr,sweaglesw@sweaglesw.org

ABSTRACT
In this paper, we propose a rule-based method to improve efficiency in bottom-up chart generation with GG, an open-source reversible large-scale HPSG for German. Following an in-depth analysis of efficiency problems in the baseline system, we show that costly combinatorial explosion in brute force bottom-up search can be largely avoided using information already contained implicitly in the input semantics: either (i) information is globally present, but needs to be made locally available to a particular elementary predication, or (ii) semantic configurations in the input have a clear translation to syntactic constraints, provided some knowledge of the grammar. We propose several performance features targeting inflection and extraction, as well as more language-specific features, relating to verb movement and discontinuous complex predicates. In a series of experiments on three different test suites we show that 7 out of 8 features are consistently effective in reducing generation times, both in isolation and in combination. Combining all efficiency measures, we observe a speedup factor of 4.5 for our less complex test suites, increasing to almost 28 for the more complex one: the fact that performance benefits drastically increase with input length suggests that our method scales up well in the sense that it effectively heads off the problem with exponential growth. The present approach of using a generator-internal transfer grammar has the added advantage that it locates performance-related issues close to the grammar, thereby keeping the external semantic interface as general as possible.

TITLE AND ABSTRACT IN GERMAN

Effiziente HPSG-Generierung für das Deutsche

Wir stellen eine regelbasierte Methode vor, zur automatischen Anreicherung der semantischen Eingabe einer reversiblen HPSG des Deutschen, die es erlaubt, teure uninformierte Suche bei der Bottom-Up-Chart-Generierung weitgehend zu vermeiden, indem (i) globale Information, die implizit in der Eingabe vorhanden ist, explizit und lokal verfügbar gemacht wird, und (ii) syntaktische Constraints aus semantischen Konfigurationen abgeleitet werden. Wir schlagen Performanzfeatures für verschiedene Phänomene vor, wie Flexion, Extraktion, Verbbewegung und diskontuierliche komplexe Prädikate. Unsere Experimente zeigen erhebliche Effizienzsteigerungen (Faktor 4.5–Faktor 27.8), deren Zunahme mit steigender Eingabekomplexität korreliert, was die gute Skalierbarkeit unserer Methode belegt. Der generator-interne Transferansatz zeichnet sich weiterhin dadurch aus, daß Performanzaspekte grammatik-nah behandelt werden, wodurch die externe Semantikschnittstelle so allgemein wie möglich bleibt.

KEYWORDS: Surface generation, HPSG, German.

1 Introduction

1.1 HPSG bottom-up generation and non-configurationality

Recent advances in the efficiency of bottom-up chart generation with reversible HPSG grammars (Carroll and Oepen, 2005), namely local ambiguity factoring under subsumption and index accessibility filtering, appear to have solved the most pressing efficiency problems associated with HPSG generation for English, turning reversible linguistically motivated grammars like the ERG (Copestake and Flickinger, 2000) into interesting resources for offline and online surface generation. While the efficiency measures implemented in the LKB and ACE generators (see section 1.2) are also effective for German, these measures appear to be insufficient to resolve generation performance issues for GG, a large-scale HPSG for German originally developed at DFKI (Müller and Kasper, 2000; Crysmann, 2003, 2005, 2007). In fact, even on moderately complex inputs, the generator quickly runs into a combinatorial explosion, having so far prevented the grammar from being usable for any serious real-time NLG tasks.

Upon closer inspection of the source of the inefficiency, it became quickly apparent that the observed performance problems are the result of a conspiracy of several factors, most of which can be subsumed under the notion of non-configurationality:

- Relatively free constituent order

  In contrast to English, constituent order in German clauses is relatively free, permitting permutation of complements, including the subject, as well as interspersal of modifiers in pretty much any position. As a result, chart size grows rather quickly, with ambiguity packing being ineffective until rather large, i.e. mostly clausal, structures are built.

- Verb placement

  German finite verbs display a placement alternation between clause-initial (V1/V2) and clause-final realisation, determined by clausal construction type. Under a bottom-up regime, both left-branching and right-branching structures must be explored. Furthermore, PPs and sentential complements easily extrapose across final verbs, thereby increasing the search space even more. In the case of particle verbs, initial placement of the verb leaves the particle in final position, giving rise to discontinuous lexical items, related by (simulated) head movement (Kiss and Wesche, 1991; Müller and Kasper, 2000; Crysmann, 2003, among others).

- Argument composition

  Auxiliaries, modals, raising verbs, and, optionally, control verbs form a verb cluster with their non-finite complements. Arguments of upstairs (=governing) and downstairs (=governed) verbs can be interleaved (e.g. ... *weil ein Buch*$_2$ *er*$_1$ *ihm*$_2$ *zu kaufen*$_2$ *versprach*$_1$ 'because he$_1$ promised$_1$ to buy him$_2$ a book$_2$.'), making it necessary to compose arguments of the downstairs verb (*kaufen* 'buy') onto the valence lists of the upstairs verb (*versprechen* 'promise'), resulting in the creation of complex predicates.

  Argument composition interacts with both free constituent order and verb placement. In particular the latter means that some members of the composed valence list must be hypothesised before the initial verb has been encountered, leading to partially underspecified valence lists (e.g. *Letzte Woche versprach*$_1$ *ein Buch*$_2$ *er*$_1$ *ihm*$_2$ *zu kaufen*$_2$ 'Last week, he$_1$ promised$_1$ to buy him$_2$ a book$_2$.'). Since the underspecified valencies are not constrained as to the identity of the argument (no semantic Skolem constants), any chart item (e.g., *letzte Woche* ‘last week’) that matches the underspecified syntactic description can sneak in (i.e., locally satisfy the hypothesised valency), thereby creating massive local ambiguity.

- Partial VP fronting

  Verb fronting in German may leave some (or all) arguments behind for realisation in the *Mittelfeld* (e.g. *Kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *ihm*$_2$ *das Buch*$_2$ *morgen]. / Das Buch*$_2$ *kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *ihm*$_2$ *morgen]. / Ihm*$_2$ *das Buch*$_2$ *kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *morgen].* ‘He$_{1,2}$ should$_1$ buy$_2$ him$_2$ the book$_2$ tomorrow.’). Since the core sentence has to be generated before it combines with the fronted element, construction of the core sentence needs to proceed without any access to valence information. As a result we experience a massive combinatorial explosion that can only be controlled very late, i.e. once the entire sentential structure has been built.

- Rich inflection

  While not a problem in itself, the fact that German NPs are inflected for case multiplies the existing performance issues, most specifically in the case of underspecified valence information, since irrelevant case inflection can only be detected quite late.

In the present paper we suggest a method that automatically enriches the input semantics in such a way, as to derive local syntacto-semantic constraints from the global semantic configuration: as a consequence we shall be able to eliminate globally unsuccessful generator hypotheses early on in bottom-up chart generation.

```
[ LTOP: h0
  INDEX: e19
  RELS: < [ prpstn_m_rel LBL: h0 MARG: h16  ARG0: e19 TPC: x17  PSV: u2 ]
   [ "_pron_n_ppro_rel" LBL: h3
     ARG0: x17 [ --TOP: + --COH: + --PUNCT: prop-punct --CAS: n-list PNG.PN: 2s ] ]
   [ "pronoun_q_rel" LBL: h7 ARG0: x17 RSTR: h9  body: h10  ]
   [ "_sollen_v_modal-haben_rel" LBL: h18
     ARG0: e19 [ --TOP: - --COH: - --SIND: 2s --PUNCT: prop-punct
                 --TPC: tpc-non-event-non-mod --SUB: -
                 TENSE: present MOOD: indicative PERFECTIVE: - ]
     ARG1: h14 ]
   [ "_schnarchen_v_n-haben_rel" LBL: h12
     ARG0: e15 [ --TOP: - --COH: - --PUNCT: prop-punct --TPC: tpc-non-event-non-mod
                 --SUB: bool TENSE: untensed PERFECTIVE: - ]
     ARG1: x17 ] >
  HCONS: < h18 qeq h16 h9 qeq h3 h14 qeq h12 > ]
```

Figure 1: Enriched input MRS for *Du sollst schnarchen* ‘You should snore’

1.2 The ACE generator

The open source ACE platform (http://sweaglesw.org/linguistics/ace/) implements a natural language generator based primarily on chart generation (Kay, 1996). The input to the generation system is a grammar and the semantics of the utterance to be generated (expressed in Minimal Recursion Semantics, or MRS (Copestake et al., 2005)). The grammar is primarily a declarative formalism, in that it defines a bidirectional relationship between MRSes and strings. The generator’s output is the list of all strings which are related to the input MRS by the grammar. To combat the exponential worst-case complexity of the chart generation algorithm, ACE deploys two key efficiency measures described by (Carroll and Oepen, 2005), namely ambiguity packing under subsumption and index accessibility filtering. In these respects it is quite similar to the LKB parser-generator system (Copestake, 2002). While ACE, just like the LKB, supports not only parsing and generation modes, but also MRS-based transfer, its main advantage resides in its processing efficiency: compared to the LKB, generation speed on the LOGON Rondane treebank (1169 items, avg. sentence length: 14.13) is 14.7 times better than that of the LKB, bringing average generation times down from 6.34s (LKB) to 0.43 (ACE).

2 MRS term rewriting for generation efficiency

The central mechanism by which we intend to address the generation efficiency problem is to automatically enrich the input semantics to the generator in such a way that global information implicitly present in the MRS representation will be made explicit and locally available on relevant elementary predications. By doing so, we will make them ultimately accessible to the grammar during generation. By means of an automated and quasi-deterministic rewrite step on the input MRS, we hope to strike a good balance between a maximally grammar-independent external semantic interface, suitable for application developers, and an enriched input to the generator that will hopefully reduce brute-force search by means of automatically derived syntacto-semantic constraints.

2.1 The LOGON MRS term rewrite system

Within the context of the LOGON MT project (Oepen et al., 2004, 2007), the LKB processing and development platform was extended with a term rewrite system for semantic representation using Minimal Recursion Semantics (Copestake et al., 2005).

In the LOGON system, a *transfer grammar* is a sequential, resource-sensitive set of rewrite rules which, when applied one after another in order, transform an MRS produced by a source-language grammar into an MRS suitable for NLG with the target-language grammar. We adopt the same formalism for a different purpose. A rewrite rule is a tuple of patterns for matching pieces of MRSes, consisting of an *input* pattern, a *context* pattern, a *filter* pattern and an *output* pattern. A rule $< I, C, O, F >$ causes a part of the current MRS matching the input pattern I to be replaced with the output O, provided that the context C also matches the input MRS and the filter F does *not* match. The patterns can be interdependent, so that e.g. the output can copy information matched by the input and the context. Each of the four patterns I, C, O, F contains any number of descriptions of elementary predications[1].

In an MRS (cf. figure 1), an elementary predication consists of a predicate name and any number of named roles, whose values are logical variables. A description of an elementary predication can use a regular expression to constrain which predications can match it. Several sorts of type constraints can be imposed on the values of individual roles, including unifiability with, subsumption by, or equality to a particular type (*x,e,u,i,h*). As for *properties*, i.e. features, of such variables, matching is restricted to mere unifiability. Finally, it is possible to specify that two or more variables matched in different elementary predication descriptions within the same rule have the same identity, by assigning a coreference tag. Similarly, for variable properties.

Formally, the rewrite system has the computational power of a Turing machine. As such, it is not possible to give bounds on the time complexity of applying an arbitrary rule set. However, in practice the operation is tractable for the types of rule sets considered here.

[1]Descriptions involving handle constraints are also possible, though less common.

2.2 Implementation of term rewrite system in ACE

The ACE platform, similarly to the LKB, allows grammarians to interpose a step of term rewriting between the declarative portion of the grammar and the publicly displayed MRS. The purpose of this is to allow grammarians additional freedom in designing the MRS schema described by the grammar proper, while maintaining a semantic interface that is more stable between grammar revisions and also affording an opportunity to remove remnants of non-semantic information. Since the term rewrite system is not bidirectional, separate rule sets are used after parsing and before generation.

The external MRS input to the generator is passed through the *pre-generation* rewrite system, resulting in a so-called internal MRS input (cf. figure 1). It is this MRS that is used to identify the initial set of grammar entities that need to be added to the chart. The immediate result of chart generation is a set of strings together with the sequence of grammar rules and lexemes that licensed them, and the MRS corresponding to that analysis. The result MRSes are passed through the *post parsing* rewrite rules, resulting in external MRSes. Only those strings whose corresponding external MRSes are subsumed[2] by the external MRS input are output.

A single rule can match an MRS multiple ways. Due to resource sensitivity, the order in which the matches have the rule applied can in principal affect the outcome. When a rule matches the input in K different places, there are $K!$ possible match orderings to try, which could each yield a different result, making the complexity of the operation worse than exponential. In practice, this issue can be so severe that the time spent in the rewriting process dominates the overall generation time. However, through careful rule-writing it is possible to ensure commutativity. ACE features a mode in which only one (arbitrary) match ordering is performed, rather than executing all $K!$ orderings (only to determine that they all have identical results). We exploit this feature to reduce the time spent in rewriting to a fraction of the overall generation time.[3]

2.3 Rule-based enrichment of input semantics

In this subsection, we shall present in some detail the individual efficiency measures our transfer grammar automatically derives from the generic external semantic representation. The efficiency measures we implemented can be largely classified into three groups: inflection-related measures, which mainly reduce the number of inflectional variants in the chart, German-specific measures related to verb placement and argument composition, and finally, extraction-related measures.

Most of the enrichment was done by means of having the transfer grammar augment semantic variables with additional performance features. Values of these features are typically atomic types (cf. figure 1 for a sample MRS: performance features are prefixed with two dashes and rendered in blue). In one case, i.e. *oind*, the transfer grammar adds an additional role argument (individual variable) to relevant elementary predications. On the grammar side, rules were additionally constrained according to these efficiency features. Unless specialised to some value by the MRS rewrite grammar, the enriched grammar rules will apply just as before, enabling us to measure performance gains by simply activating or deactivating blocks of transfer rules.

[2]By subsumption of MRSes, it is meant that every predicate in the input MRS must be realised in the output MRS, and the identity of the logical variables is the same (modulo renaming). It is considered permissible for the output MRS to be more specific than the input MRS, permitting, inter alia paraphrase generation by input underspecification.

[3]Average transfer processing times on the three test suites discussed in this paper are as follows: MRS: 13.2ms, TSNLP: 12.1ms, Babel: 29.4ms. Transfer times are already included in the overall processing time reported in table 1 below.

In this subsection, we shall discuss each measure in turn, together with brief remarks on the implementation and an estimate of the expected benefits.

2.3.1 Inflection

Case (*cas*) One of the most straightforward efficiency measures to come up with when confronted with a highly inflectional language such as German is to eliminate inflected forms from the chart that cannot possibly be part of a globally well-formed realisation. While some inflectional forms are readily filtered by the semantic input or the lexicon, namely predicate-inherent information such as TAM (tense/aspect/mood) and number/gender for nominal expressions, this is not the case for morphosyntactic case, which is determined by properties of the governing predicate.

In a configurational language, the expected inefficiency of inflecting every NP for all possible cases, even irrelevant cases may be suboptimal, but not really a matter for concern, since the NP will locally combine with its governing predicate, rendering NPs in irrelevant cases inert during further search. In a non-configurational language such as German, which features argument composition, heads combine with complements that are not their own arguments but rather those of a predicate they compose with, i.e., they need to cater for unknown raised arguments by means of underspecified valence lists. As a result, the identity of the inherited arguments is not known, so any XP present in the chart, however inflected, can sneak into these underspecified lists, to be ruled out, in the majority of cases, only when a significantly larger structure has been built. Unfortunately, languages with an articulate case system tend to be of the non-configurational, rather than the configurational type.

In order to predict the case for NPs, we developed a set of 35 rules that derive case requirements from lexical and structural properties of the input semantics (cf. the `--CAS` feature in figure 1). While some cases are indeed trivial, e.g., predicting the case of obliques, or arguments of prepositions, others are not: first, since individuals can be arguments to more than one predicate, as witnessed in relative clause constructions, such individual variables must be exempted from case prediction. Second, raising and, in particular, voice alternation can change case assignment properties. Thus, the transfer grammar must carefully anticipate these properties in a case by case fashion.

Apart from an overall slight reduction in chart size, we expect this feature to be particularly useful in all constructions involving locally underspecified valence lists, including the quite common case of separable particle verbs and raising and control constructions, as well as more specific, yet quite expensive ones like partial VP fronting.

Punctuation (*punct*) The implementation of punctuation in GG (Kilian, 2007) follows that of the English ERG in using inflectional rules. Even when limiting ourselves to basic sentence punctuation (commas, period, question mark, exclamation mark), almost every chart item can be inflected in 5 different ways, given that it cannot be known a priori which chart item will end up, e.g., at the right periphery of the entire sentence, where sentence mode (declarative/interrogative/imperative) is expressed. What is globally known, however, is sentence mode: all it takes is to distribute exactly this information onto every elementary predication (cf. `--PUNCT` in figure 1). Abstracting away from quotations, every sentence will be in only one of the three modes, so the number of punctuation variants for each chart item can be brought down to 3 (instead of 5).

This measure, while simple-minded and straightforwardly implemented (5 rules), is nevertheless expected to be highly efficient, given that it indiscriminately targets almost every lexical edge (heads and dependents alike) and therefore has a significant impact on the overall size of the search space, given the bottom-up regime of the generator. The only edges that do not benefit from this (or any other measure discussed in this paper) are those corresponding to semantically empty lexical items, like, e.g., auxiliaries, relative pronouns etc.

2.3.2 Verb movement and direction of branching (sub)

A peculiarity of German syntax that has quite strong repercussions on processing efficiency is verb placement: while non-finite verbs are placed in phrase-final position, finite verbs display an alternation between final and initial position: in relative clause, embedded interrogatives, as well as subordinate clauses introduced by a complementiser or subordinating conjunction, the finite verb is realised in final position, otherwise initially, including matrix clauses.

The global construction type ("matrix order" vs. "subordinate order") can be calculated from properties of the input semantics, in particular, by taking into consideration sentence mode (declarative vs. interrogative), the kind of embedding (relativisation, complementation, type of conjunction), as well as the presence and nature of the topicalised element (embedded V2 vs. embedded wh vs. that-clause). The pre-generation transfer grammar uses this information to determine for each verb whether it is in a "subordinate" or "non-subordinate" context (cf. `--SUB` in figure 1). In addition to predicting direction of branching for simple verbs, the main benefit of this feature is that we can decide when to hypothesise head movement of the verb.

2.3.3 Discontinuous complex predicates

Coherent vs. non-coherent constructions (*coh*) Probably one of the strongest factors responsible for generation inefficiency is due to discontinuous complex predicates leading to local underspecification of valence lists that permit sneaking in from any XP edge in the chart, triggering massive combinatorial explosion. Fortunately, whether some predicate permits argument composition or not is a lexical matter, with composition being restricted to auxiliaries, modals, raising and control verbs. Thus, the transfer grammar marks the arguments of non-finite complements of modals etc., as to their potential of undergoing argument composition (*coh* +). Likewise, arguments of finite verbs that are expressed periphrastically (perfective, future, passives) are marked for composition. With arguments of all other predicates being marked with a negative value, underspecified valence lists can be protected to some degree against illicit intrusion of arguments (cf. `--COH` in figure 1). This feature is expected to be particularly helpful in those cases where we are confronted with entirely underspecified valence lists, as with separable particle verbs and partial VP fronting.

Predicting upstairs objects (raising/control) As discussed in the introduction, discontinuous verb clusters may necessitate hypothesising valencies of the initial verbs to be realised in the *Mittelfeld*, in particular objects of initial raising and control verbs that intersperse with the arguments of the final verb or verb cluster. Since the subcategorisation requirement of the upstairs verb are not known during bottom-up construction of the *Mittelfeld*, additional arguments are hypothesised even in cases where the initial verb takes no complement at all. Furthermore, such hypothesised argument slots provide potential for illicit intrusion.

On the basis of the predicate argument structure, however, it is quite straightforward to decide not only whether an argument should be hypothesised or not, but also to determine its identity. To this end, the transfer grammar redundantly encodes the upstairs verb's object as an additional argument role (*oind*) which is used by the grammar to restrict any additional argument slot.

Predicting properties of raised subjects (*sind*) The last feature relating to complex predicates targets modals and subject raising verbs, which agree with a subject that is not their own argument. In order to limit the number of inflected variants of potentially expensive items in the chart (they all trigger argument composition), the transfer derives the person-number information of the syntactic subject from the argument structure of their non-finite argument (cf. `--SIND` in figure 1). Since the scope of this feature is limited, we did not have any a priori expectation as to whether the potential gains in certain construction will be sufficient to offset the overhead incurred by the extended rule set.

2.3.4 Non-local dependencies

Long distance dependencies, like topicalisation, wh-fronting and relativisation are a notorious source of inefficiency in syntactic processing. In German, extraction is very common: even in ordinary declaratives, some constituent is extracted from the matrix or an embedded clause and placed into the sentence-initial *Vorfeld*, a kind of topic position. In the external MRS, the distinguished individual variable of the topicalised element is represented as an information-structural property of the proposition or question relation (TPC feature). Topicalised elements can be arguments, modifiers (scopal or intersective), as well as heads, in the case of (partial) VP fronting. Moreover, as a side-effect of wide-spread scrambling, there is no canonical position even for arguments, let alone adjuncts, so gap prediction is vital.

Local vs. non-local realisation (*top*) Predicting local vs. non-local realisation of arguments is expected to be both straightforward and effective: given that the individual variable of the topicalised element is already registered in the external MRS, it is almost sufficient to redundantly encode this fact as a property of the variable, thereby making it visible on the governing predicate as well, i.e., the context from which extraction proceeds, and mark all remaining arguments as local (cf. `--TOP` in figure 1). This basic scenario gets, of course, slightly more complicated given the fact that individual variables can be arguments of more than one predicate, which may or may not be a reason for concern: in the case of across-the-board extraction from coordinate structures, it can be harmless, whereas in the case of relativisation, we are confronted with the possibility of an individual which is realised locally with respect to the upstairs predicate, yet non-locally within the relative clause. In the transfer grammar, this is resolved by means of a three-valued system of types (+,−,*na*), where Boolean values correspond to topicalised (+) and non-topicalised (−) realisation, whereas *na* represents the neutral case (relativisation). Both local head-complement rules (*na_or_−*) and complement extraction rules (*na_or_+*) are made sensitive to this distinction.

As a consequence of this feature, we expect some considerable reduction in chart size: with the exception of relativised arguments, all arguments will be marked as either local or non-local, thereby eliminating a great deal of non-determinism.

Predict extraction type and gap site (*tpc*) While prediction of argument extraction as sketched above can reduce some of the complexity incurred by long-distance dependencies, it is rather moot when it comes to items that are not represented on the argument structure of the local head at all, e.g. modifiers.

Taking into consideration the semantic relation that the topicalised elementary predication enters in with some other elementary predication, it is possible to detect, from the external

MRS, both modifier status (scopal vs. intersective) and location of the gap site, i.e. the modified item. In a similar way, it is possible to identify cases of (partial) VP fronting.

Complementing the *top* feature, which marks extraction as a property of arguments, the transfer grammar introduces a feature *tpc* to identify the locus of the gap as a property of the head. Values of this feature serve to distinguish further between different types of extraction, e.g. intersective vs. scopal modification, verb fronting and plain argument extraction (cf. `--TPC` in figure 1). Extraction rules are made to be sensitive to properties of the head, accordingly. The ordered transfer rules first try to detect instances of modifier fronting and partial VP fronting and mark the event variable of the elementary predication corresponding to the head for the appropriate extraction type. All remaining verbal and predicative elementary predications are marked as a potential site for argument extraction, thereby ruling out modifier or verb fronting from these sites.

A priori, it is difficult to assess the efficiency of this feature. However, given the rather unconstrained nature of modification, and therefore modifier extraction, it is safe to expect some decent benefit.

3 Evaluation

In order to assess the impact of the proposed measures on generation efficiency, we carried out several experiments on three different regression test suites for German: the Babel test suite (Müller, 2004), the TSNLP test suite (Lehmann et al., 1996), and the German version of the CSLI MRS test suite. The test suites were parsed, and successfully analysed test items were subsequently disambiguated using the Redwoods treebank annotation tool (Oepen et al., 2002). This left us with a total of 2,259 semantic input representations for the generator (Babel: 609, TSNLP: 1547, MRS: 103).

None of the test suites used in the experiments here was specifically designed for the purposes of NLG. Rather, all three are general purpose, phenomenon-oriented regression test suites. However, there are some differences in the design of the individual test suites that we expect to affect the impact of our performance improvements: while the MRS and TSNLP test suites consist of rather short utterances (MRS: 4.44 words/item, TSNLP: 4.76 words/item), Babel is slightly more complex (6.76 words/item). Another important difference relates to the kind of phenomena included in the test suite: to give an example, TSNLP includes a fair amount of non-sentential items for testing NP-internal agreement, a phenomenon which should be entirely unaffected by most of the efficiency measures suggested here, which are all targeted at clausal syntax.

All test runs were performed on a Linux (kernel 2.6.32) compute server with 12 Intel Xeon X5650 2.67GHz CPUs and 16GB RAM, running 4 processes in parallel (on an otherwise idle machine). The ACE generator was run in standard configuration, i.e. with a memory limit of 1.2 GB for forest creation plus another 300 MB for unpacking. The number of realisations per item was limited to 1000. Tests have been profiled using [incr tsdb] (Oepen, 2002).

In addition to comparing the performance of the full pre-generation transfer grammar to that of the baseline, we conducted a number of additional test runs to evaluate the effectiveness of each performance feature on its own (*+feature*), as well as each feature's contribution to the combined performance (*−feature*).

3.1 Results

The main results are summarised in tables 1 and 2, giving crucial performance indicators (overall processing time and passive edges) for all three test suites. Comparing baseline performance (*Base*) to the combined effect of all features (*All*), we observe a speedup of around a factor of 4.5 for MRS and TSNLP test suites. On the more complex Babel test suite efficiency gains even go up to a factor of almost 28.[4] As indicated in table 1, average speedup on all three test suites is at 18.5.

	MRS		TSNLP		Babel		MRS+TSNLP+Babel	
	Time (s)	Red.	Time (s)	Red.	Time (s)	Red.	Time (s)	Red.
Base	0.257	1.00	0.273	1.00	7.213	1.00	2.143	1.00
+cas	0.222	1.16	0.216	1.26	4.547	1.59	1.384	1.55
+punct	0.150	1.71	0.159	1.71	4.598	1.57	1.355	1.58
+sub	0.210	1.22	0.215	1.27	6.042	1.19	1.786	1.20
+coh	0.226	1.14	0.275	0.99	5.681	1.27	1.730	1.24
+oind	0.242	1.06	0.262	1.04	5.506	1.31	1.675	1.28
+sind	0.262	0.98	0.274	1.00	7.035	1.03	2.096	1.02
+top	0.130	1.97	0.136	2.00	5.309	1.36	1.530	1.40
+tpc	0.131	1.96	0.178	1.53	3.602	2.00	1.099	1.95
All	0.054	4.71	0.063	4.31	0.260	27.79	0.116	18.52
-cas	0.056	4.59	0.065	4.20	0.371	19.46	0.147	14.58
-punct	0.063	4.11	0.081	3.38	0.415	17.39	0.170	12.61
-sub	0.058	4.44	0.067	4.09	0.304	23.71	0.130	16.44
-coh	0.054	4.78	0.061	**4.49**	0.353	20.45	0.139	15.41
-oind	0.056	4.59	0.062	4.37	0.526	13.72	0.187	11.46
-sind	0.050	**5.12**	0.062	4.40	0.258	**27.95**	0.114	18.74
-top	0.076	3.37	0.081	3.37	0.381	18.92	0.162	13.25
-tpc	0.064	4.01	0.079	3.46	0.897	8.04	0.299	7.17
-index	0.049	5.24	0.069	3.97	0.433	16.67	0.166	12.91
-pack	0.079	3.23	0.087	3.14	0.563	12.82	0.215	9.98

Table 1: Processing time and speedup factor

Similarly positive efficiency factors can be observed regarding space consumption (edges), although space savings typically fall short of time savings, given the fact that we are using a generator with ambiguity packing.[5]

Table 2 also details generation coverage achieved by each test run: on MRS and TSNLP test suites, coverage is 100% throughout. On Babel, we achieve full coverage, once a sufficient number of efficiency measures is enabled (second half of table 2). Test runs for baseline performance, as well as those with only a single performance feature activated at a time (top half of table 2), occasionally run into memory exhaustion, accounting for reduced coverage. However, since coverage on all test runs is either greater or equal to that of the baseline, a potential floor effect benefits the baseline more than any other runs, thus leaving the significance of our results unaffected.

[4]Apparently, combination of features pays off much better in terms of time savings than mere multiplication of individual factors would suggest, an effect that has been previously noted in the context of chart generation (Carroll and Oepen, 2005).

[5]Passive edges reported in table 2 are packed edges: thus, any edge filtered by our performance features can lead to time savings at several points during generation, namely edge creation, packing, and unpacking.

	MRS			TSNLP			Babel		
	Cov	Edges	Red.	Cov	Edges	Red.	Cov	Edges	Red.
Base	100.0	703	1.00	100.0	693	1.00	96.7	4864	1.00
+cas	100.0	663	1.06	100.0	630	1.10	97.9	3817	1.27
+punct	100.0	440	1.60	100.0	458	1.51	97.5	3364	1.45
+sub	100.0	555	1.27	100.0	567	1.22	96.9	4137	1.18
+coh	100.0	645	1.09	100.0	692	1.00	98.0	4460	1.09
+oind	100.0	675	1.04	100.0	676	1.02	98.2	3959	1.23
+sind	100.0	706	1.00	100.0	692	1.00	96.9	4859	1.00
+top	100.0	392	1.79	100.0	441	1.57	98.0	3539	1.37
+tpc	100.0	410	1.71	100.0	452	1.53	98.0	2931	1.66
All	100.0	116	6.07	100.0	172	4.03	100.0	554	8.78
-cas	100.0	133	5.28	100.0	194	3.57	100.0	693	7.02
-punct	100.0	179	3.92	100.0	242	2.87	100.0	860	5.65
-sub	100.0	141	4.99	100.0	195	3.55	100.0	673	7.23
-coh	100.0	124	5.67	100.0	175	3.96	100.0	657	7.41
-oind	100.0	121	5.82	100.0	174	3.99	100.0	841	5.78
-sind	100.0	117	6.00	100.0	172	4.03	100.0	558	8.72
-top	100.0	205	3.43	100.0	251	2.76	100.0	825	5.90
-tpc	100.0	150	4.68	100.0	228	3.04	100.0	1173	4.15
-index	100.0	126	5.58	100.0	198	3.50	100.0	682	7.13
-pack	100.0	215	3.27	100.0	292	2.37	99.7	1292	3.77

Table 2: Generation coverage and space consumption (passive edges)

Investigating the impact of the individual features in more detail, we find that almost all of them are effective on at least two of the three test suites. The only exception is *+sind*, the feature which calculates subject agreement information for raising and modal verbs: not only do we not find any clear benefits in isolation; its inclusion also proves detrimental in combination with other features. Given that this measure is highly specific, its failure to give rise to positive effects is hardly surprising. All other features are effective not only in isolation (top half of the table), but we can observe from the runs in the lower half of each table (leave-one-out) that each feature still has an impact when used in combination.

Starting with the inflection-oriented features, i.e., *cas* and *punct*, we find that both have consistent impact on all test suites. However, the effect of controlling punctuation is clearly stronger than that of predicting case: in fact, punctuation is the second to third most effective feature of all features tested. We believe that this is due to the following factors: first, predicting morphosyntactic case can only ever have an effect on nominal expressions (nouns, determiners, attributive adjectives), whereas punctuation will affect every lexical item in the chart that corresponds to some elementary predication in the input, targeting nominal and non-nominal expressions alike. Furthermore, while case prediction only reduces the number of potential complements, predicting punctuation also has an effect on heads and modifiers. Second, sentential punctuation is a global feature, i.e., all elementary predicates will be specialised in the same way. Case assignment, however, is a local property, and must therefore cater for the situation where an individual variable is shared by two or more governing predicates, as, e.g., in the case of relativisation. As a net effect, individual variables must be exempt from case prediction in these cases. This explanation is further supported by the fact that the reduction factor on passive edges is very close to the theoretically maximal value for punctuation of 1.67 (5/3).

The features related to verb placement and argument composition (*sub,oind,coh*) all lead to performance improvements, albeit to differing degrees, depending on the feature and the test suites: while prediction of verb placement, i.e., direction of branching and presence/absence of verb movement (*sub*), leads to consistently good effects on all three test suites, as does the prediction of the absence/identity of the initial verb's object (*oind*), the *coh* feature, which exclusively caters for argument composition, shows more variable behaviour: though beneficial otherwise, space savings on TSNLP are negligible, with processing times even going up slightly. This may not be too surprising: while prediction of verb placement and direction of branching affect every clause, the *coh* feature will only show an effect in constructions involving particular predicates or tenses, which is a situation that can vary depending on the concrete input.

Finally, the two extraction features *top* and *tpc* show again consistent and highly effective performance improvements across all test suites, both in isolation and in combination. This confirms quite neatly our initial expectation that these two efficiency measures are largely independent, the former (*top*) targeting complement extraction, by virtue of their being represented on the head's argument structure, the latter (*tpc*) targeting the remaining cases, most notably gap prediction for adjunct extraction. Finally, the fact that long distance dependencies are not only costly, but also frequent and not specific to any particular construction, explains why good gap prediction gives rise to consistently high performance benefits across all test suites.

Before we close the presentation of the main results, we would like to briefly compare the efficiency of the measures proposed here to those suggested by Carroll and Oepen (2005), namely index accessibility filtering and ambiguity packing. Disabling each of these previously established performance features in turn (cf. the last two rows in tables 1 and 2), we can show that the detrimental effect shown on our test data is comparable to that incurred by disabling one of the features investigated here.

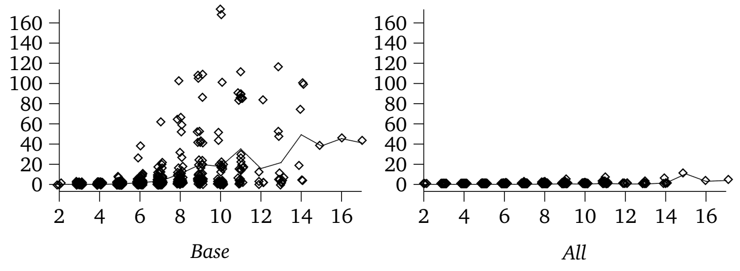

Figure 2: Processing time (in s) per string length (babel)

3.2 Discussion

We have observed during the presentation of the main results that the majority of MRS-derived efficiency features show comparable speedup effects across the three different test suites, when used in isolation. Notable exceptions were the somewhat more construction and, therefore, input-specific features *coh* and *oind* which are highly dependent on the presence of complex predicates. However, we observed quite strong differences (a factor around 5.5) as to the cumulative effects between babel on the one hand and the less complex TSNLP and MRS test suites on the other. In order to better understand the significance of the experiments reported on here we shall investigate the differences and discuss what practical implications will ensue.

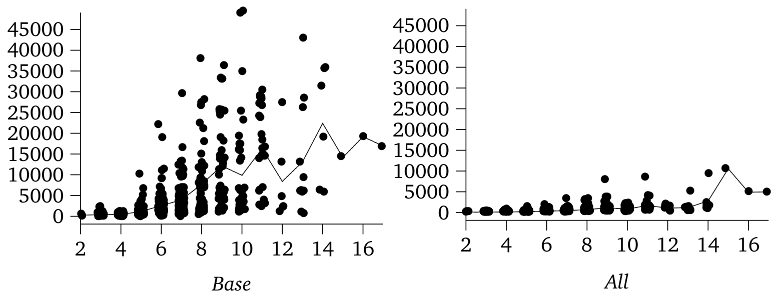

Figure 3: Passive edges per string length (babel)

The most obvious difference between Babel and the other two test suites is of course input length: by artificially reducing average input length on Babel to slightly above (4.85) that of TSNLP (by filtering out longer inputs), the cumulative speedup factor reduces to a factor around 9.5, compared to 27.5, which is still not fully comparable, yet much closer to the factor of 4.5 observed for MRS and TSNLP suites.[6] The impact of input length on relative generation speedup is also corroborated by the scatter plots of time and space (passive edge) consumption shown in figures 2 and 3: without any of the efficiency measures proposed in this paper, processing time begins to explode already at an average sentence length of 8 (see figure 2), averaging at around 8.4s. With the efficiency measures, processing time never even comes close to that level, leading to massive performance gains on longer inputs. The comparison of passive edges in figure 3 confirms even more clearly how the current efficiency measures particularly counter the combinatorial explosion observable with the baseline. To summarise, while all test suites witness good reduction of average processing times, it is clear that the real benefit of the generation efficiency measures suggested here becomes apparent with longer (and therefore more complex) inputs. For practical purposes, in particular for online processing, taming of the worst case complexity for longer inputs is more important than speedup factors on the relatively short utterances characteristic of MRS and TSNLP test suites.

Before we close, we should like to address the issue of how our method could be ported to grammars for languages other than German: while some of the concrete features we used are somewhat specific to German (or Dutch), others should be easily portable. The punctuation feature, as well as the the two extraction-related features *top* and *tpc* should be useful to improve generation efficiency in a wide range of languages: in a small experiment carried out with the ERG (Copestake and Flickinger, 2000) on the Rondane treebank (see section 1.2), we observed a 10.5% reduction in generation time for punctuation alone. We expect that other features, such as the *cas* feature will be useful for other less configurational languages, such as Slavic languages, given the fact that elaborate case systems and relatively free word order often go hand in hand.

3.3 Related work

An alternative strand of approaches towards efficient processing of unification grammars builds on the idea of compiling these grammars into formalisms with better worst-case complexity

[6]Reducing average string length to slightly below that of the MRS test suite (4.37), results in a cumulative speedup factor of only 5.4.

than native unification-based processing, such as CFG or TAG: e.g., Kiefer and Krieger (2000) proposed a CFG superset approximation of HPSG for parsing English and Japanese. However, this method has so far never been successfully applied to German, let alone for generation. Furthermore, despite potentially better raw performance, CFGs are plagued by at least as severe locality issues as bottom-up HPSG generation. TAGs, by contrast, with their extended domain of locality, constitute a much more interesting target formalism for compiling an HPSG generation grammar. In the literature, two such approaches have been reported: Kasper et al. (1995) describe a method of compiling Klaus Netter's HPSG of German to TAG, but the compilation did not cover the full grammar but only a fragment, and, unfortunately, no performance measures are reported for either parsing or generation. Becker and Lopez (2000) specifically capitalise on the fact that TAG's extended domain of locality gives rise to an a priori expectation towards greater generation efficiency, and, building on Kasper et al. (1995), they describe a compilation of the Verbmobil English and Japanese HPSG grammars into LTAG. Again, however, no performance tests are reported that could substantiate the claim of increased generation performance with the compiled grammar. Furthermore, to the best of our knowledge, no such compilation has ever been carried out for German.

In the context of native unification-based processing, Gardent and Kow (2007) suggest a method to enrich the semantic input to an FTAG of French with tree features that permit almost deterministic selection of generation paraphrases. Moreover, Gardent and Kow (2005) argue that such selection also leads to performance improvements, as they show on the basis of sample sentences. With respect to the German LFG (Rohrer and Forst, 2006; Cahill et al., 2007), Zarrieß and Kuhn (2010) propose a transfer approach to provide f-structure input for the XLE surface realiser from shallow semantic representations. The main motivation for this was that f-structures contain a high level of syntactic and even morphosyntactic detail that make them less suitable for paraphrasing and, more generally, for deployment in natural language generation systems. Zarrieß and Kuhn (2010) also discuss the impact of grammatical function prediction from semantic roles for generator efficiency: depending on the complexity of the transfer rules, they observe considerable differences in average generation time, ranging from 246.14s for the "naive rules" to 36.2s for their "informed rules", which operate on configurations rather than individual roles. Nakanishi et al. (2005) propose a beam search approach to tackle generation efficiency for an English HPSG. We believe their approach to be complementary to ours, since MRS enrichment can prune the search space with certainty and without locality restrictions, such that a future system using both methods will be able to provide good results at smaller beam sizes, taking advantage of a division of labour between transfer-based treatment of non-local and probabilistic pruning of local dependencies.

Conclusion

We have proposed a method to improve generation efficiency with GG, a reversible HPSG of German. Using a term rewrite system integrated into the generator, we automatically enrich the purely semantic input representation with additional syntacto-semantic constraints, derived from the semantic configuration. Evaluating our method on three different regression test suites for German, we have shown that this approach is highly successful in taming combinatorial explosion in bottom-up chart generation, leading to significant speedup factors: while on less complex inputs, we achieved a speedup by a factor of around 4.5, performance gains increase considerably on more complex inputs, yielding a speedup factor of almost 28, which shows that our method scales up well to increasing input lengths.

References

Becker, T. and Lopez, P. (2000). Adapting HPSG-to-TAG compilation to wide-coverage grammars. In *Proceedings of the 5th International Workshop on Tree-Adjoining Grammars and Related Formalisms (TAG+5)*, pages 47–54, Paris.

Cahill, A., Forst, M., and Rohrer, C. (2007). Stochastic realisation ranking for a free word order language. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 17–24, Saarbrücken, Germany. DFKI GmbH. Document D-07-01.

Carroll, J. and Oepen, S. (2005). High efficiency realization for a wide-coverage unification grammar. *Natural Language Processing–IJCNLP 2005*, pages 165–176.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford.

Copestake, A. and Flickinger, D. (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.

Copestake, A., Flickinger, D., Pollard, C., and Sag, I. (2005). Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3(4):281–332.

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

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

Crysmann, B. (2007). Local ambiguity packing and discontinuity in German. In Baldwin, T., Dras, M., Hockenmaier, J., King, T. H., and van Noord, G., editors, *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Gardent, C. and Kow, E. (2005). Generating and selecting grammatical paraphrases. In *Proceedings of the 10th European Workshop on Natural Language Generation*.

Gardent, C. and Kow, E. (2007). A Symbolic Approach to Near-Deterministic Surface Realisation using Tree Adjoining Grammar. In *45th Annual Meeting of the Association for Computational Linguistics - ACL 2007*, pages 328–335, Prague, Czech Republic. Association for Computational Linguistics.

Kasper, R., Kiefer, B., Netter, K., and Vijay-Shanker, K. (1995). Compilation of HPSG to TAG. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 92–99, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kay, M. (1996). Chart generation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 200–204. Association for Computational Linguistics.

Kiefer, B. and Krieger, H.-U. (2000). A context-free approximation of Head-Driven Phrase Structure Grammar. In *Proceedings of the 6th International Workshop on Parsing Technologies (IWPT 2000)*, pages 135–146.

Kilian, N. (2007). Zum Punkt gekommen. Master's thesis, Universität des Saarlandes, Saarbrücken.

Kiss, T. and Wesche, B. (1991). Verb order and head movement. In Herzog, O. and Rollinger, C.-R., editors, *Text Understanding in LILOG*, number 546 in Lecture Notes in Artificial Intelligence, pages 216–240. Springer-Verlag, Berlin.

Lehmann, S., Oepen, S., Regnier-Prost, S., Netter, K., Lux, V., Klein, J., Falkedal, K., Fouvry, F., Estival, D., Dauphin, E., Compagnion, H., Baur, J., Balkan, L., and Arnold, D. (1996). Tsnlp - test suites for natural language processing. In *The 16th International Conference on Computational Linguistics (COLING)*, volume 2, pages 711–716, Copenhagen, Denmark. Center for Sprogteknologi.

Müller, S. (2004). Continuous or discontinuous constituents? A comparison between syntactic analyses for constituent order and their processing systems. *Research on Language and Computation*, 2(2):209–257.

Müller, S. and Kasper, W. (2000). HPSG analysis of German. In Wahlster, W., editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 238–253. Springer, Berlin.

Nakanishi, H., Miyao, Y., and Tsujii, J. (2005). Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 93–102. Association for Computational Linguistics.

Oepen, S. (2002). *Competence and Performance Profiling for Constraint-based Grammars: A New Methodology, Toolkit, and Applications*. PhD thesis, Saarland University.

Oepen, S., Callahan, E., Flickinger, D., Manning, C., and Toutanova, K. (2002). LinGO Redwoods: A rich and dynamic treebank for HPSG. In *Beyond PARSEVAL. Workshop at the Third International Conference on Language Resources and Evaluation, LREC 2002*, Las Palmas, Spain.

Oepen, S., Dyvik, H., Lønning, J. T., Velldal, E., Beermann, D., Carroll, J., Flickinger, D., Hellan, L., Johannessen, J. B., Meurer, P., Nordgård, T., and Rosén, V. (2004). Som å kapp-ete med trollet? Towards MRS-based Norwegian–English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD.

Oepen, S., Velldal, E., Lønning, J. T., Meurer, P., Rosén, V., and Flickinger, D. (2007). Towards hybrid quality-oriented machine translation. On linguistics and probabilities in MT. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, Skövde, Sweden.

Rohrer, C. and Forst, M. (2006). Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of LREC-2006*, Genova.

Zarrieß, S. and Kuhn, J. (2010). Reversing f-structure rewriting for generation from meaning representations. In Butt, M. and King, T. H., editors, *Proceedings of the LFG10 Conference, Ottawa*, pages 479–499, Stanford. CSLI publications.