

Approaching ambiguity in real-life sentences – the application of an Optimality Theory-inspired constraint ranking in a large-scale LFG grammar

Jonas Kuhn and Christian Rohrer

Institut für maschinelle Sprachverarbeitung, Universität Stuttgart

Azenbergstraße 12, D-70174 Stuttgart, Germany

{kuhn,rohrer}@ims.uni-stuttgart.de

1 Introduction

It is well-known that the ambiguity problem is one of the key problems on the route from experimental “toy grammar” encodings to industrial-size linguistic systems designed for application: for almost any non-trivial sentence, the required complex rule system will produce a multitude of readings. No less obvious has been the realization that there is no easy once and for all answer to the problem. Attempts to cope with it range from linguistic constraining on various levels (syntax, semantics) over context- and world knowledge-based inferencing to statistical filtering; for certain applications like Machine Translation, it has also been argued that often (though not always), disambiguation can be avoided. For different recent approaches to the paradigm case for ambiguity – PP attachment – see the contributions in Mehl *et al.* (1996).

The work reported in the present paper – the use of constraint ranking within unification grammars, basically inspired by work in the Optimality Theory literature (OT, cf. e.g., Prince and Smolensky (1993); Bresnan (1996a,b)) – does not come with the ambition to provide an unheard-of solution to the general ambiguity problem itself. The main objective is rather to shift the attention to some practical considerations that help to do the groundwork – the engineering of large-scale grammars – with minimal interference of the ambiguity problem. Marked constraints are used to deal with both local ambiguities and ambiguities that survive on the sentence level. This scheme used to filter ambiguity during grammar development may or may not be used for disambiguation in applications of the grammar.

One of the general engineering policies adopted in the ParGram project¹ is to avoid commitment to early design decisions in ill-understood areas (here, the decision for a certain non-optimal disambiguation scheme) that may influence a number of other decisions and components.

The straightforward way to pursue this strategy is to put off all filtering of readings, i.e. deal with the full number of readings for any sentences during development. This is however impractical in grammar engineering for at least two reasons: (i) longer sentences can have such a great number of readings, originating from different types of ambiguity, that inspection by the grammar writer is very time-consuming, even if state-of-the-art inspection tools are utilized; (ii) ambiguity poses an efficiency problem, even for the best available parsing algorithms using

¹In this project of parallel grammar development, large-scale grammars for English, French and German are being developed at Xerox PARC in Palo Alto, California, the Rank Xerox Research Centre in Grenoble, France, and the Institut für maschinelle Sprachverarbeitung, Stuttgart; cf. <http://www.ims.uni-stuttgart.de/projekte/pargram/>.

packed representations (cf. e.g. Maxwell and Kaplan (1996); Dörre (1997)) – in this light, it is desirable in engineering to be able to “switch off” the effect of ambiguities irrelevant to the analysis under consideration in order to isolate its effects and minimize the run time of test suites. What is required is a filtering mechanism that can be easily re-organized, enforced or switched off altogether.

The system context for the reported study is a large-scale encoding of an LFG grammar² of German, using the Xerox Linguistic Environment (XLE), which is being developed by Ron Kaplan, John Maxwell and colleagues. One main objective in grammar design is to pursue linguistically well-motivated analyses while at the same time responding to efficiency and maintainability requirements. The idea is that a solid linguistic foundation is a guarantee for continuity and consistency in development (given linguistically well-trained grammar writers). This situation affects the choice of filtering strategies during grammar development. The linguist should be able to fully understand and control the filtering scheme.

As an ideal candidate for such a scheme, we propose the use of constraint ranking as it has been proposed in the linguistic literature within the field of OT. The particular advantage of such a move in our context is that such a constraint ranking combines most advantages of a weighting approach to disambiguation (Stolcke (1993); Briscoe and Carroll (1993); Carroll and Rooth (1996)) with symbolic control by the grammar writer over the means of expression. In section 2, we briefly introduce the underlying mechanism of constraint ranking, as provided by the XLE platform. Section 3 illustrates its use with some simple instances of ambiguity. In section 4, the merits of the scheme for a systematic methodology in grammar engineering are discussed. Section 5 finally mentions some further uses and prospects of constraint ranking.

2 Constraint ranking inspired by Optimality Theory

The constraint ranking used here is not a direct implementation of a certain version of OT from the theoretical literature. The XLE system rather provides an extension of the most common ranking mechanism used in OT. In order to be able to rank LFG constraints, optimality marks have to be introduced in the grammar. This is done by constraints that refer to an *o*-projection (the *o*-structure being a flat multiset of optimality marks percolated from all daughter constituents).

$$(1) \quad S \rightarrow \left\{ \begin{array}{c} A \\ \text{MARK1} \in o^* \\ \\ B \\ \text{MARK2} \in o^* \end{array} \right\} C$$

In (1) e.g., *S* expands to *AC* or *BC*, alternatively. In the former case, *MARK1* gets introduced to the *o*-structure, and *MARK2* in the latter.

The ranking of the optimality marks is specified in the header of a grammar. The standard OT ranking of dispreferred constraints is extended to a scheme that allows both dispreference and preference.

(2) OPTIMALITYRANKING MARK1 MARK2 NEUTRAL MARK3 MARK4 NOGOOD MARK5.

²Lexical-Functional Grammar, cf. Bresnan (1982); Dalrymple *et al.* (1995).

In (2), MARK1 is more preferred than MARK2; all unmarked constraints are treated as NEUTRAL; MARK3 is dispreferred, MARK4 is even more dispreferred. MARK5 finally, may not show up in any successful solution, since it is to the right of the keyword NOGOOD; an analysis containing such a mark is eliminated right away (early in the constraint propagation step of parsing).

The set of remaining analyses in a parse are compared and filtered according to the following criteria: First of all, the most dispreferred mark is checked (MARK4 in example (2)); the analyses with the *smallest* number of this mark survive. If there is more than one analysis left, the next dispreference mark (MARK3) is considered etc. In case the dispreference marks do not favour a unique candidate, the preference marks are also considered, starting from the most preferred one (MARK1). Here, the analyses with the *greatest* number of this mark win. Again, the marks next in the ranking (MARK2) are considered if necessary. If the lowest-ranked preference mark does not produce a unique solution, a set of analyses is output.

Marks that are introduced in the grammar, but do not show up in the ranking are treated as NEUTRAL. This is very useful in testing the grammar by systematic variation (cf. section 4 below).

3 Filtering of ambiguity

The constraint ranking scheme provides the prerequisites to write a full grammar based on this OT variant. In the present context however, it is merely used as an add-on utility for an existing large LFG grammar of German. By convention, grammaticality distinctions are being realized by hard (i.e. non-ranked) constraints; the ranking is used only to filter implausible readings in the case of ambiguity, according to the objective laid out in the introduction.

The use of constraint ranking for this purpose is relatively straightforward. Optimality marks have to be put on the disjunctive part of the grammar that gives rise to a certain ambiguity. Depending on what is intended, one disjunct may be marked by a preference marker or (another one) by a dispreference marker.

For illustration, we will go through an example that takes care of the subject-object ambiguity, which is quite frequent in German.

(3) Anna sucht Otto.

In (3), there is no overt morphological case marking of nominative and accusative, resulting in ambiguity in terms of the function of the NPs.

This may be filtered by introducing a preference for a subject NP in the *Vorfeld* (simplified extremely):

$$(4) S \rightarrow \left\{ \begin{array}{l} \text{NP} \\ \left[\begin{array}{l} (\uparrow\text{SUBJ}) = \downarrow \\ \text{VORFELDSUBJ} \in o^* \end{array} \right] \\ (\uparrow\text{OBJ}) = \downarrow \end{array} \right\} \begin{array}{l} \text{VP} \\ \uparrow = \downarrow \end{array}$$

Assuming that the mark VORFELDSUBJ is ranked as a preference mark (as opposed to a dispreference mark), the system will output for (3) only the reading in which *Anna* is the subject of the verb *sucht*.

In order to get the right analysis for an example like (5) – where despite word order and the missing overt case marking, *Otto* is the preferred subject – we could use an additional preference based on selectional restrictions of the verb.

(5) Das Fahrrad reparierte Otto.

Assume we have the following lexicon entry for *reparierte*:

(6) *reparierte* V (↑PRED) = ‘repair((↑SUBJ),(↑OBJ))’
 (↑SUBJ) =_c NOM
 $\left\{ \begin{array}{l} \left[\begin{array}{l} (\uparrow\text{SUBJ ANIM}) =_c + \\ \text{SELRESTRANIM} \in o^* \end{array} \right] \\ (\uparrow\text{SUBJ ANIM}) \neq + \end{array} \right\}$
 (↑OBJ) =_c ACC

When the disjunct requiring the subject to be animate (ANIM +) is chosen in an analysis, the mark SELRESTRANIM is introduced to the o-structure. Under a ranking that gives a higher preference to this mark than to the VORFELDSUBJ mark, (5) will receive the correct analysis. One may ask why the selectional restriction is not expressed as a hard constraint. This is to avoid having to tell a complete story about polysemous subjects like in *die Stuttgarter Werkstatt reparierte das Fahrrad* at this point, according to the policy of avoiding an unflexible early design decision where possible.

A natural usage of a dispreference mark is given in the following example. The existence of elliptical NPs like *die mittelgroßen* makes it necessary to allow for a headless type of NP. This leads to ambiguity in sentences like (7):

(7) Hier fehlen gemütliche Kneipen.

Since the verb *fehlen* takes an optional dative object, and *Kneipen* is morphologically ambiguous between nominative and dative, the sentence has an extra (implausible) reading with *gemütliche* as the subject and *Kneipen* as the object.

We may exclude this by marking the use of a noun-less NP with a dispreference mark:

(8) NP → (Det) A $\left\{ \begin{array}{l} \text{N} \\ \text{e} \\ \text{EMPTYNOUN} \in o^* \end{array} \right\}$

Concluding this section, here is the optimality ranking assumed in the examples discussed:

(9) OPTIMALITYRANKING SELRESTRANIM VORFELDSUBJ NEUTRAL EMPTYNOUN NO-GOOD.

4 Constraint ranking and grammar engineering

From the previous section, it should be clear how the constraint ranking mechanism can be exploited to filter different kinds of ambiguity in a large unification grammar during development. For certain ambiguities, this may be even an appropriate long-term treatment. For other cases, like the PP attachment ambiguity, it is not likely that one can find an effective way of tying disambiguation to grammatical constraints. However, as we said in the introduction, this is not intended anyway.

The merits of the reported scheme lie rather in the support of a very flexible, but still systematic methodology of engineering and, in particular, internal evaluation. The basic idea is sketched in this section.

A major problem in the development of large-scale (unification) grammars is the complex interaction. It is hard to detect the exact source of an interaction problem, and even harder to find a remedy against a known problem. The use of optimality marks now supports an easy and error-proof testing of fine-grained variants of the grammar.

In this context, the marks are used to remove certain parts of the grammar by writing the mark beyond the NOGOOD limit. A similar effect is created traditionally by commenting out a certain rule or part of a rule. However, commenting out has some drawbacks: there is always a risk of errors when commenting out and back in again, especially if there are nested comments; also, it is hard to keep track of several places in the grammar that one wants to vary systematically in a controlled comparison; one frequently forgets to reactivate a bit of code. Finally, there is no direct way to re-try a certain variation after some time; one has to figure out again which region to comment out.³

Since the optimality ranking is located in the header of the grammar, systematic, error-proof variation can be achieved with very little effort. It is very straightforward to write test routines that perform the changes automatically and run a test suite over a large number of slightly different grammar versions. On this basis, the interaction problem can be tackled with some realistic prospect of success.

The table in fig. 1 gives a short passage from a test result as an illustration. The phenomenon under consideration is the elliptical NP, as already discussed in section 3, and in particular, its interaction with other phenomena.

The columns of the table represent the test results for different version, where the two numbers for each sentence are the number of readings and the parse time in seconds. Grammar version a. treats all constraints as NEUTRAL, version b. activates the constraint ranking, treating the empty noun head as dispreferred (thus, the filtering of readings for test items (iii-iv)). Version c. has the mark EMPTYNOUN moved to the right of NOGOOD, which causes a change in coverage – (i) no longer gets a parse. In version d., an additional NP variant is excluded: bare demonstrative pronouns that are identical with the article like *die* in (ii).

The table reveals that there is an interaction problem: when the noun-less NP is de-activated (c.), the parsing of longer sentences with a great number of NPs (involved in attachment ambiguities) becomes considerably faster (v-ix), although the number of parses remains unaffected. The effect of blocking NPs like *die* (d.) has an even more drastic effect. The reason

³The problems can be avoided when some macro processor is used that allows for conditional compilation, as it is known from programming languages. Still, we consider the optimality mark-based scheme proposed below as superior, since it can be directly combined with the filtering function discussed in section 3 (re-positioning a mark on the left of NOGOOD), and furthermore makes use of the same syntax as the LFG specification language (with the *o*-projection), such that errors are recognized by the system.

		a. no ranking	b. ranking	c. *empty N	d. *dem. pronoun *empty N
(i)	er sucht die mittelgroßen.	1 0.442	1 0.548	0 0.299	0 0.275
(ii)	die gefallen ihr.	1 0.305	1 0.305	1 0.463	0 0.194
(iii)	in der Stadt fehlen gemütliche Kneipen.	2 0.483	1 0.473	1 0.450	1 0.384
(iv)	in der Stadt fehlen die schönen kleinen angenehmen gemütlichen Kneipen.	6 5.404	1 5.471	1 1.248	1 0.938
(v)	er sieht das Kind.	1 0.312	1 0.301	1 0.334	1 0.278
(vi)	er sieht das Kind mit der Mütze.	2 0.673	2 0.673	2 0.696	2 0.395
(vii)	er sieht das Kind mit der Mütze in der Hand.	5 1.505	5 1.516	5 1.603	5 0.594
(viii)	die Erfahrungen sollen später in die künftigen Planungen für die gesamte Stadt einfließen.	2 12.625	2 14.944	2 4.542	2 1.186
(ix)	hinter dem Betrug werden die gleichen Täter vermutet, die während der vergangenen Tage in Griechenland gefälschte Banknoten in Umlauf brachten.	92 217.418	20 222.776	20 35.580	20 4.632

Figure 1: Results of a controlled comparison based on marked constraints

is roughly that, with such NPs being allowed, a lot of hopeless NP candidates are postulated in the parsing process that can be excluded only quite late.

While corpus examples like (viii) and (ix) are adequate for detecting the presence of an interaction problem, the above test results cannot be straightforwardly used to analyze the exact nature of the problem, which is a precondition for finding a fix. To aid the problem analysis, the proposed constraint marking technique can be combined with systematic methods of test suite construction: a set of test items is composed to bring out details of the interaction problem. A short sample test suite is given in the appendix. In fig. 2, a graphical representation of the test results for this test suite is given for the grammar versions b. through d.

Test items 1-20 focus on singular NPs containing a determiner. In the unrestricted grammar (b.), the cost for computing PP attachment grows faster, the more adjective attributes there are (compare 1-5 to 6-10, 11-15 and 16-20, respectively). Although no successful analysis of the examples will contain a noun-less NP, the rule allowing such a construction is responsible for the growing complexity, as the comparison with the graph from grammar version c. shows – here additional adjectives don't result in slower parsing.

The other point to note about the test results is the following: in the unrestricted grammar version b., and in version c., there is a considerable difference between parsing times for singular NPs, containing a determiner (1-20), and for determinerless plural NPs (21-40). The comparison with version d. reveals that this is mainly due to the possibility of hypothesizing bare demonstratives whenever there is a definite article.

Now, what does this problem analysis help us? We certainly cannot disallow the ‘bad guys’ –

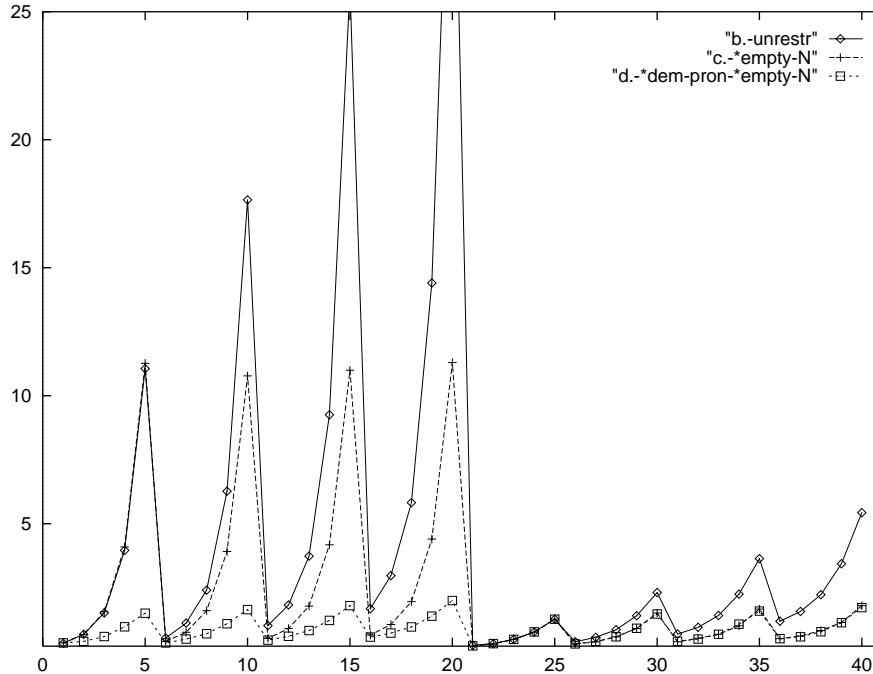


Figure 2: Systematic evaluation of interaction effects (x-axis: examples from the test suite in the appendix; y-axis: runtime in sec.)

noun-less NPs and bare demonstratives – in the main grammar. But the experiments with the grammar variant show quite clearly where a more careful restriction of the rules (causing an earlier exclusion of candidates) is worthwhile. In effect, insights from the reported experiment have led to a revised analysis with considerably improved runtime behaviour – e.g., sentence (ix) from fig. 1 is parsed in 34.559 seconds, although the coverage of the grammar is equivalent to version b., rather than restricted as in versions c. and d. The details of this improvement are however beyond the scope of the present paper whose focus has been on engineering methods and diagnostic tools rather than a particular grammatical analysis.

5 Conclusion

We have proposed a scheme of using constraint ranking effectively in the engineering of large unification grammars. This methodology has been applied and is being extended in the ParGram project for the LFG grammar of German.

Further applications of the OT-inspired constraint ranking in this context include the treatment of non-core phenomena. These can be covered by rules that introduce dispreference marks, such that they become relevant only when no core-grammar analysis is available for a certain utterance.

Once the set of phenomena that can be successfully treated by the ranking scheme discussed in section 3 has stabilized, the automated test routines proposed in section 4 can be applied to induce the relative ranking of the constraints from larger annotated test suites and text

corpora. In this context, ranking could also be applied to the ambiguity types we excluded from our treatment, like PP attachment. Thus, the mainly practically motivated use of a ranking in unification grammars can effectively be one step on the route towards a deeper treatment of ambiguity in general.

Appendix: The interaction test suite

1. er sieht das Kind.
2. er sieht das Kind mit der Mütze.
3. er sieht das Kind mit der Mütze in der Hand.
4. er sieht das Kind mit der Mütze in der Hand vor dem Haus.
5. er sieht das Kind mit der Mütze in der Hand vor dem Haus mit dem Turm.
6. er sieht das kleine Kind.
7. er sieht das kleine Kind mit der Mütze.
8. er sieht das kleine Kind mit der Mütze in der Hand.
9. er sieht das kleine Kind mit der Mütze in der Hand vor dem Haus.
10. er sieht das kleine Kind mit der Mütze in der Hand vor dem Haus mit dem Turm.
11. er sieht das nette kleine Kind.
12. er sieht das nette kleine Kind mit der Mütze.
13. er sieht das nette kleine Kind mit der Mütze in der Hand.
14. er sieht das nette kleine Kind mit der Mütze in der Hand vor dem Haus.
15. er sieht das nette kleine Kind mit der Mütze in der Hand vor dem Haus mit dem Turm.
16. er sieht das nette kleine blonde Kind.
17. er sieht das nette kleine blonde Kind mit der Mütze.
18. er sieht das nette kleine blonde Kind mit der Mütze in der Hand.
19. er sieht das nette kleine blonde Kind mit der Mütze in der Hand vor dem Haus.
20. er sieht das nette kleine blonde Kind mit der Mütze in der Hand vor dem Haus mit dem Turm.
21. er sieht Kinder.
22. er sieht Kinder mit Mützen.
23. er sieht Kinder mit Mützen in Gärten.
24. er sieht Kinder mit Mützen in Gärten vor Häusern.
25. er sieht Kinder mit Mützen in Gärten vor Häusern mit Türmen.
26. er sieht kleine Kinder.
27. er sieht kleine Kinder mit Mützen.
28. er sieht kleine Kinder mit Mützen in Gärten.
29. er sieht kleine Kinder mit Mützen in Gärten vor Häusern.
30. er sieht kleine Kinder mit Mützen in Gärten vor Häusern mit Türmen.
31. er sieht nette kleine Kinder.
32. er sieht nette kleine Kinder mit Mützen.
33. er sieht nette kleine Kinder mit Mützen in Gärten.
34. er sieht nette kleine Kinder mit Mützen in Gärten vor Häusern.
35. er sieht nette kleine Kinder mit Mützen in Gärten vor Häusern mit Türmen.
36. er sieht nette kleine blonde Kinder.
37. er sieht nette kleine blonde Kinder mit Mützen.
38. er sieht nette kleine blonde Kinder mit Mützen in Gärten.
39. er sieht nette kleine blonde Kinder mit Mützen in Gärten vor Häusern.
40. er sieht nette kleine blonde Kinder mit Mützen in Gärten vor Häusern mit Türmen.

References

- Bresnan, J. W., ed. (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA, first edition.
- Bresnan, J. W. (1996a). Optimal Syntax: Notes on Projection, Heads, and Optimality. Ms., Stanford University.
- Bresnan, J. W. (1996b). LFG in an OT Setting: Modelling Competition and Economy. In *Proceedings of the First LFG Conference*. CSLI Proceedings ON-LINE.
- Briscoe, T. and Carroll, J. (1993). Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars. *Computational Linguistics*, **19**(1), 25–59.
- Carroll, G. and Rooth, M. (1996). Valence induction with a head-lexicalized PCFG. Ms., Institut für maschinelle Sprachverarbeitung, Stuttgart.
- Dalrymple, M., Kaplan, R. M., Maxwell III, J. T., and Zaenen, A., eds. (1995). *Formal Issues in Lexical-Functional Grammar*. CSLI Publications, Stanford, CA.
- Dörre, J. (1997). Efficient Construction of Underspecified Semantics under Massive Ambiguity. Ms., Institut für maschinelle Sprachverarbeitung, Stuttgart.
- Maxwell III, J. and Kaplan, R. (1996). An Efficient Parser for LFG. In *Proceedings of LFG Conference 1996*.
- Mehl, S., Mertens, A., and Schulz, M., eds. (1996). *Präpositionalsemantik und PP-Anbindung*, vol. 16 of *Schriftenreihe Informatik*. Uni-GH Duisburg, Institut für Informatik.
- Prince, A. and Smolensky, P. (1993). Optimality Theory: Constraint Interaction in Generative Grammar. Technical Report 2, Rutgers University Center for Cognitive Science, Piscataway, NJ.
- Stolcke, A. (1993). An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities. Technical Report TR-93-065, Internat. Computer Science Inst., Berkeley, CA.