Resolving some apparent formal problems of OT Syntax

Jonas Kuhn
Universität Stuttgart

Resolving some apparent formal problems of OT Syntax

Jonas Kuhn

Universität Stuttgart

0. Introduction

In this paper,[1] I present a formalization of Optimality Theoretic Syntax and address some apparent formal and computational problems that such a comparison-based syntax model has to face. The main focus is on a set of problems having to do with the complexity of the processing tasks (parsing or recognition, and generation) when working with an OT grammar. The strategy adopted is to restrict the expressiveness of the formalism in a way that is compatible with the linguistic intuitions to be modelled. As it turns out, this move will also solve other, conceptual problems. The individual problems are:

A. (complexity problem:) When the comparison-based definition of grammaticality in Optimality Theory (OT) is applied literally in a computational model of syntax, a massive complexity problem arises: for the determination of grammaticality, a large (potentially infinite[2]) set of highly complex candidate analyses has to be computed and checked for constraint violations (cf. also Johnson 1998 for a decidability problem).

Two aspects of the complexity problem can be distinguished: (i) What restrictions can be imposed on the formalism to make the processing tasks decidable, i.e., to guarantee that algorithms can be devised that perform the tasks? (ii) Can further restrictions guarantee that the processing complexity lies in a realistic complexity class?

[1] Parts of this paper are abbreviated sections from Kuhn 1999.

[2] Results by Tesar (1995:ch. 2,3) show that infinity of the candidate set is not a problem by itself: his dynamic programming techniques can deal with infinite candidate sets. The challenge for a computational treatment of OT Syntax is to come up with algorithms for grammars beyond context-freeness, and without the input being restricted to a "strictly ordered sequence of elements" (Tesar 1995:111).

NELS 30

Aspect (i) is discussed at length in Kuhn 1999, here I will present just a summary and report on work in progress on aspect (ii).

B. (ambiguity problem:) According to Smolensky (1996), comprehension and production do not differ in terms of the algorithm applied for selecting the most harmonic analysis—in production the candidates share the same underlying representation, in comprehension the same surface form. Hale and Reiss (1998) point out that this is incompatible with the ubiquitous phenomenon of ambiguous surface forms (it is predicted that only the most harmonic reading should exist).

C. (conceptual problem:) To account for language-particular ineffability, Legendre, Smolensky, and Wilson (1996) assume that a candidate unfaithful to the input LF (or Index LF) can win a competition. Even under the "inventory perspective" on OT that they adopt, the intuitive status of such LF-unfaithful analyses, which duplicate another—faithful—analysis without meaning anything different, is obscure. An alternative that Legendre et al. (fn. 8) discuss would be to assign no interpretation at all to LF-unfaithful winners. This seems more plausible, but would require some additional mechanism (applying after harmony evaluation), which checks the optimal candidate's LF against the input/Index LF, and nullifies the result in case of divergence.

In this short paper I cannot go into the technical details of problem C. (one reason being that the account involves a different set of assumptions than the OT model adopted here to tackle problem A.). Nevertheless, the discussion of complexity issues in OT syntactic processing opens up the possibility for a new view on the status of LF-unfaithful winners: The constraint-compilation framework of sec. 5 below can be seen as an implementation of the inventory idea, and the LF-unfaithful winners "disappear" from the online processing model.

Sec. 1 of this paper introduces the non-derivational model of OT syntax proposed by Bresnan (1998), sec. 2 discusses the formalization of constraints. In sec. 3, the processing task of parsing, besides generation, and the complexity issue is addressed. In sec. 4, I propose to exploit relative locality of the domain of constraint interaction. Sec. 5 finally sketches how this idea might be taken even further in an offline compilation approach to optimization.

1. Candidate generation in a non-derivational model of OT syntax

Many notions of OT Syntax are still subject to discussions, so it would be too early to fix them once and for all in a formalization. Nevertheless it is important for this paper to be able to pinpoint a particular conception for certain notions formally, such that, e.g., computational consequences can be investigated.

I will follow the OT syntax model proposed by Bresnan (1998, a.o.),[3] which is based on the formalism of Lexical-Functional Grammar (LFG) and has been called the

[3] Bresnan shows that Grimshaw's (1997) analysis can be reconstructed in the non-derivational framework. Most examples in the present paper are adopted from this fragment.

OT-LFG system. Adopting this approach has the advantage that results from the rich computational literature on LFG can be applied. LFG's system of correspondence between parallel structures[4] turns out very useful in the OT context.

The input for candidate generation that Bresnan assumes is "a (possibly underspecified) feature structure representing some given morphosyntactic content independent of its form of expression" (Bresnan 1998:sec. 1.1). An example (that in English would have *we saw you* as its optimal realization) is given in (1).

(1) Input f-structure

$$\begin{bmatrix} \text{PRED} & \text{'SEE}\langle x, y\rangle\text{'} \\ \text{GF}_1 & \begin{bmatrix} \text{PRED} & \text{'PRO'} \\ \text{PERS} & 1 \\ \text{NUM} & \text{PL} \end{bmatrix}_x \\ \text{GF}_2 & \begin{bmatrix} \text{PRED} & \text{'PRO'} \\ \text{PERS} & 2 \\ \text{NUM} & \text{SG} \end{bmatrix}_y \\ \text{TNS} & \text{PAST} \end{bmatrix}$$

In a fully specified f-structure, concrete instantiations of the grammatical functions GF_i will appear: SUBJ, OBJ etc.; also, further features may be added. The candidate analyses in an OT competition consist of such fully specified f-structures, together with a c-structure that realizes the underlying information. Some such c-structures for (1) are given in (2).

(2) a. $[_{\text{IP}}$ we $[_{\text{I}'}$ did $[_{\text{VP}}$ see you$]]]$
b. $[_{\text{IP}}$ we $[_{\text{I}'}$ saw $[_{\text{VP}}$ you$]]]$
c. $[_{\text{CP}}$ saw $[_{\text{IP}}$ we $[_{\text{I}'}$ $[_{\text{VP}}$ you$]]]]$

The c-structure/f-structure pairings have to obey inviolable principles that can be formalized as a "classical" LFG grammar: According to the "extended head theory" adopted in recent LFG work (see Bresnan 1999), all positions in the extended X-bar-schema are optional, including the heads. This freedom is controlled by f-structural and c-structural well-formedness principles. The system allows for a strictly non-derivational conception of the base grammar (without having to assume traces at c-structure).

With this underlying grammar of inviolable principles (which we can call G_{Inviol}), candidate generation can be formalized as follows:

[4] The most important levels of representation are c(ategorial)-structure and f(unctional)-structure. Principles about the former can be captured by context-free rules, the latter is constrained by feature equations based on c-structure categories. A correspondence function ("projection") maps categories on f-structures—typically as a many-to-one function.

(3) For a given input f-structure Φ_i, the set of candidate analyses $Gen(\Phi_i)$ is the set of LFG (c- and f-structural) analyses $\langle T, \Phi' \rangle$ generated by G_{Inviol}, such that Φ_i subsumes Φ'.

Hence, the task of determining the candidate set for a given input consists of monotonically adding information to the input f-structure, plus finding a corresponding c-structure. This is exactly the classical task of generation from a semantic structure.

As Wedekind (1999) shows, this problem is undecidable in the general case. However, as discussed in Kuhn 1999:sec. 3.1, it is plausible to assume formal restrictions on allowable grammars and on the potential to add f-structure information during generation that will avoid the undecidability problem.[5] This methodological move is typical for the whole enterprise that the present contribution is a part of: linguistic accounts applying a certain formalism attempt to arrive at explanatory restrictions of the full power of the formalism. Likewise, for a computational account certain restrictions have to be assumed that will guarantee decidability, or membership in a certain complexity class. The interesting question is whether these two motivations for restricting the formalism converge in some way or other.

2. Constraint marking and harmony evaluation

OT constraints are typically formulated as universally quantified implications over structural elements, saying that whenever a structure satisfies description *A*, it should also satisfy description *B* (simpler constraints can be generalized to this form). An obvious reaction would be to express the constraints as formulae in a logic with appropriate quantifiers, and model the constraint marking function *marks* as checking whether a given candidate structure satisfies these general formulae.

To allow for multiple violations, structure checking has to continue even when an inconsistence has arisen. This creates an indeterminacy for formulae involving several quantifiers, which is somewhat counterintuitive: For instance, what does the hypothetical constraint (4) say about a candidate containing a DP that has two non-nominal daughters, like (5)? Does (5) violate (4) once or twice? Both might be plausible.

(4) *Hypothetical constraint*
For all DP categories, all their daughters are nominal.
$\forall n.[\mathrm{DP}(n) \rightarrow \forall m.[M(n,m) \rightarrow nom(m)]]$

(5) ... $[_{\mathrm{DP}}$ V PP] ...

The formulae (6) and (7) are both equivalent to (4) in terms of classical logic, but they yield different results for our candidate (here intuitions are fairly clear): (5) should violate (6) twice, but (7) only once.

[5] The restriction has consequences in particular for the generation of "unfaithful" candidates. Input information cannot be deleted from the candidate analysis. However, Bresnan's (1998) lexicalist approach to faithfulness violations is compatible with this restriction (see Kuhn 1999:sec. 3.3).

(6) Every daughter of a DP category is nominal.
$\forall m.[\exists n.[M(n,m) \wedge \mathrm{DP}(n)] \rightarrow nom(m)]$

(7) For every category, if it has a non-nominal daughter, then it is not a DP.
$\forall n.[\exists m.[M(n,m) \wedge \neg\, nom(m)] \rightarrow \neg\, \mathrm{DP}(n)]$

So, assuming fully general formulae to model constraints is problematic. Certainly, more work on the logic of violable constraints is required, but we may introduce a rather simple restriction on the form of violable constraints that avoids the problem and that is also in line with the goal of keeping OT constraints simple (as is argued for example in Grimshaw 1998). The idea is to take out of the constraint formulation the explicit generalization over structures (which makes the constraint recursively applicable on all embedded structures). Instead, the universal applicability is now implicit to all constraints and will be made effective in the checking routine that the candidate structures have to undergo after they have been constructed: at every structural object (category/f-structure), all constraints are applied. This application of the constraints to multiple objects is the only source for multiple violations—a single structural element can violate each constraint only once. The constraints are then interpreted classically.

In order for this to work, the structural object which a given constraint focuses on has to be clearly identified. I will assume a metavariable $\star$ for this (reminiscent of the * used in f-annotations of categories to refer to the category node itself). When the constraints are checked, the metavariable $\star$ will be instantiated to one structural element after the other. Thus, the constraints are actually specified as constraint schemata, generating classical constraints when instantiated.

We can now express (4) in either of the following two ways in (8a), bringing out the difference in constraint violations incurred by a candidate like (5) much more clearly: (8a) is about DPs and is thus violated once by (4); (8b) is about daughters of DPs, so (4) incurs two violations.[6]

(8) a. $\mathrm{DP}(\star) \rightarrow \forall m.[M(\star, m) \rightarrow nom(m)]$
b. $\forall n.[\mathrm{DP}(n) \rightarrow (M(n,\star) \rightarrow nom(\star))]$

It is compatible with the restriction on constraint formulation to assume a "scalar" interpretation of alignment constraints like, e.g., HEAD LEFT (9). (Under a scalar interpretation, this constraint is violated twice if there are two intervening elements between a (single) head and the left periphery of its projection.) The metavariable-based formulation allows for a clear distinction between the non-scalar and the scalar version of this constraint, as shown in (10) (it is assumed that the function *proj* and the relations *M* —for mother—and *precede* are defined appropriately).

[6] A more intuitive way of expressing (8b) is presumably (i), which is equivalent (cf. also (6)). Note that with the constraint schemata, classical equivalences can be exploited again.
(i) $\exists n.[M(n, \star) \wedge \mathrm{DP}(n)] \rightarrow nom(\star)$

(9) HEAD LEFT: (Grimshaw 1997:374)
The head is leftmost in its projection.

(10) non-scalar interpretation: $head(\star) \rightarrow \neg\exists n.[M(proj(\star), n) \wedge precede\ (n, \star)]$
scalar interpretation: $cat(\star) \rightarrow \neg\exists n.[head(n) \wedge M(proj(n), \star) \wedge precede(\star, n)]$

The first formulation is stated from the point of view of the head; since the instantiated schema is interpreted classically (i.e., incurring maximally one constraint violation for each structural element), a given head can violate this constraint only once (even if there are several intervening nodes to the left of it). The second formulation is from the point of view of the intervening category; thus if there are several of them, the overall structure will incur several violations of this constraint.

With constraints expressed as such schemata, constraint marking on a single candidate is an easy computational task (model checking). Since furthermore, the set output of *Gen* can be characterized by a context-free grammar (an unpublished result by Ron Kaplan and Jürgen Wedekind, p.c., 1999), harmony evaluation is indeed feasible even if there are infinitely many candidates: it suffices to check the constraint profile of a finite number of candidates, since all structures beyond a certain limit are guaranteed to be less harmonic than the candidates already checked.[7] (For some more discussion, see Kuhn 1999:sec. 3.2.)

This completes the OT model of input-based candidate generation, constraint marking, and harmony evaluation to determine the optimal, and thus grammatical candidate; so it seems that the decidability aspect of the complexity problem (A.) can be solved with some plausible restrictions on the formalism (of course, more work is needed to pinpoint the *necessary* assumptions).

3. Generation and parsing

So far, the LFG grammar capturing the inviolable constraints has been applied in the generation direction, modeling language production and capturing that only grammatical (i.e., optimal) candidates are ultimately output. It would be nice if the same set-up of determining a set of candidates, plus applying constraint marking and harmony evaluation could be applied in the opposite direction to model understanding (cf. Smolensky 1996).

However, as also Johnson (1998) observes, an OT competition among candidates with on a common phonological string does not model the standard recognition task for the language generated by a grammar (here, a grammar is an entire OT model with a particular constraint hierarchy). In particular, the string-based competition will fail to reject ungrammatical candidate strings, since all competitors share that string, so the most harmonic one will be among them. The string language accepted will in effect be the language generated by the base grammar modeling the inviolable principles.

[7] Tesar (1995) employs a similar technique to deal with infinitely many candidates in the context of regular and context-free base grammars ("G_{Invial}").

A related problem is observed by Hale and Reiss (1998): the system would predict that there are practically no ambiguous surface strings, since whenever two analyses of a string have a different constraint profile, one should be ruled out being unoptimal.

Clearly, a simple string-based competition is inappropriate to model the recognition task.[8] The reason is that (for the systems we are looking at) grammaticality is defined by optimization in the generation direction. Thus, more care has to be taken to ensure that the right candidate set enters the competition—even when the processing direction for the overall system is turned around.

A possible method is proposed in Kuhn 1999:sec. 4.1. Initially, the input string is parsed by the base grammar G_{Inviol}. The resulting parsing analyses are required only to find out possible f-structures. From these f-structures the amount of information that forms an OT input is extracted. Next, a "backward generation" step is performed, generating candidates from the extracted OT input f-structures, applying the generation-based optimization (as discussed here in sec. 1 and 2). For each of these competitions there are two possibilities: (i) the string of the optimal candidate is different from the original input string—this means that the string we started from is not grammatical for that input; or (ii) the optimal candidate is an analysis of the original string—this means we have found one grammatical analysis. If case (ii) occurs for none of the competitions based on inputs extracted from the parsing results, then the string is not contained in the language generated by the OT grammar. The approach certainly captures that many strings are ambiguous (several parsing analyses turn out to be instances of case (ii)).

A possible conceptual objection to this mechanism is that the understanding task is no longer symmetrical to the production task. A further problem is the considerable processing complexity added by the backward generation step. But even without this additional factor, the processing complexity seems problematic.

So, although restrictions can be imposed on the formalism that ensure decidability of the generation and parsing tasks, the other aspect of problem A. remains: the processing model outlined follows the conceptual definition of candidate generation quite literally. In order to determine grammaticality, all alternative realizations are effectively constructed and evaluated. With the highly unrestricted underlying grammar of inviolable principles, the processing expense is immense. Intuitively, it does not seem very plausible that generation of entire candidate structures has to take place during online processing.

The question arises if there are further possibilities of imposing plausible restrictions on the formalism in order to allow for more efficient processing. The explicit construction of global syntactic candidate analyses seems to miss some concept of

[8] This does not mean that the simple parsing task has no status in the theory. For instance, it plays an important role in the learning algorithm as robust interpretive parsing (Tesar and Smolensky 1998). Moreover, Frank, King, Kuhn, and Maxwell (1998) apply a similar parsing-based constraint system to impose a preference ranking on analyses in large computational LFG grammars (with a classical concept of grammaticality).

relative locality inherent to theoretical OT accounts.[9] In sec. 4, I will attempt to pinpoint this intuition in a more formal way. Furthermore, one intuitively expects that once the language-particular ranking is known, there should be a fairly direct way of determining the optimal candidate for a given string or semantic representation. Can't one anticipate which candidates will be losers, thus avoiding their construction in the first place? In finite-state OT phonology, a compilation of the competition is possible (with certain limitations as to multiple constraint violations) (Frank and Satta 1998, Karttunen 1998). In sec. 5, I address very briefly how this conception might be extended to syntax, building on the locality restriction of sec. 4.

4. Locally restricted OT competition

The apparent reason why candidate analyses of considerable size have to be constructed prior to optimization (which will typically rule out all but one analyses) is the following: Unlike with hard constraints, in OT one cannot discard an analysis on the basis of a local constraint violation, since the analysis may still be the best of all possible ones due to more highly ranked constraints.

Nevertheless, a striking property of OT systems in the literature is the relatively restricted structural domain to which the competition can be limited. Let us look at an example which demonstrates this relative locality: tableau (11) illustrates the competition underlying the combination of a matrix and an embedded clause (the constraint definitions are given in (12)).

(11)

			*LEX-F	AGR	FULL
a.	I not think	that he not smokes		*!*	
b.	I don't think	that he not smokes		*!	*
c.	I think not	that he not smokes	*!	*	
d.	I not think	that he doesn't smoke		*!	*
⇒ e.	I don't think	that he doesn't smoke			**
f.	I think not	that he doesn't smoke	*!		*
g.	I not think	that he smokes not	*!	*	
h.	I don't think	that he smokes not	*!		*
i.	I think not	that he smokes not	*!*		

(12) *LEX-F: No lexical heads in functional categories. (Bresnan 1998:(54))
AGR: A subject and its predicate in c-structure agree.
(i.e. A c-structure subject requires that its sister constituent have an agreeing extended head.) (Bresnan 1998:(25))
FULL(-INT) (roughly:)'Parse', i.e., use all morpholexical constraints.
(cf. Bresnan 1998:42ff)

[9] Frank and Satta (1995) also reach the conclusion—based on complexity considerations for OT phonology—that optimization should be local to structural domains of bounded complexity.

What is striking about tableau (11) is that it wouldn't have been necessary to construct the full amount of candidate analyses. Rather, it would have been enough to determine the different types of analysis each of the two clauses can adopt and compute locally for these which is the most harmonic (as in tableaux (13) and (14)). Combining the two winning local analyses will have the desired effect (leading to candidate (11e)).

(13)

				*LEX-F	AGR	FULL
	a.	I not think	CP		*!	
⇒	b.	I don't think	CP			*
	c.	I think not	CP	*!		

(14)

			*LEX-F	AGR	FULL
	a.	that he not smokes		*!	
⇒	b.	that he doesn't smoke			*
	c.	that he smokes not	*!		

The question is what properties a tableau has to have in order for such a split to be possible. A relevant formal property is defined in (15). Let us assume we have ways of identifying and composing subparts of LFG analyses (like the partial c- and f-structures corresponding to the matrix clause and the embedded clause in (11), respectively).

(15) Call two sub-structures R, S of an analysis A **harmonically independent or h-independent** with respect to a definition of *Gen* and a given set of constraints, if

(i) corresponding substructures R_i, S_l can be uniquely identified in any of the candidate analyses A_l (presumably with reference to the structure of the input), such that any distinctive constraint violation incurred by A_i is also incurred by either R_l or S_i (but not both);

(ii) the different possible substructures R_j, S_j combine freely (i.e., the candidates of the overall structure can be formed by taking the cross-product of the substructures).

(16) *Lemma*

For h-independent substructures, the computation of the overall winner is equivalent to computing the individual winners (the most harmonic R_j, and the most harmonic S_k) plus putting the substructures together.

Kuhn 1999:sec. 5.1 contains a proof sketch of lemma (16). Clearly, OT analyses have to assume some structural domains the elements in which are *not* h-independent. The whole point of the violability assumption—that even grammatical analyses may violate certain constraints—is to allow for situations in which the lesser of two evils is chosen. If the violation of some constraint C_1 is accepted in the optimal solution, then only to avoid a violation of some other constraint C_2, which would have been even worse. The part of the structure that violates C_1 cannot be h-independent from the part that might

have violated C_2. In order to evaluate the competition we have to take all combinations (local to the non-h-independent portion of structure) into account.

However, as the example (11) (and its factorization into (13) and (14)) suggested, at some level h-independent substructures may nevertheless arise. When dealing with substructures that have this property, a potentially exponential amount of processing expense could be saved: rather than constructing the cross-product of possible analyses for the individual substructures, all but the optimal candidate for the subparts can be discarded. (This presupposes that the competition for each of the substructures can be processed individually.)

It seems that most syntactic OT accounts can be captured if the domain of competition is restricted to what is contained in one extended projection (cf. Grimshaw 1991). With Bresnan's (1998, 1999) LFG account of extended projections, we have a simple way of relating the input—an underspecified f-structure—to the relevant extended projections: the lexical head of the extended projection and the functional co-head(s) are all projected to the same f-structure. Thus, already the structure of recursive embedding in the input (f-structure) determines in first approximation the domains, local to which realization alternatives will compete. So it is linguistically plausible to assume the following restriction on the interpretation of constraints:

(17) *Locality of constraint interpretation*
If a part of a structure violates a constraint and there exists an alternative realization for the underlying input not violating this constraint, then the alternative realization lies within the same extended projection.

5. Exploiting locality in processing

Having limited the scope of OT constraints to extended projections, the particular way in which h-independence is exploited in processing is still open. A conceivable option of modifying the parsing routine with "backward generation" discussed in sec. 3accordingly might be the following: the input string is parsed as usual, determining potential underlying forms conveyed by that string. For each reading, the portion of the f-structure analysis that makes up the OT-input is filtered out. Now, rather than generating entire candidate analyses for these predicate-argument-structures while leaving the optimizing competition until the end, a separate substructure for each extended projection is constructed based on the respective part of the predicate-argument-structure, and a local OT competition is computed, passing on just the winning candidate. (Under a head-first processing regime, this should allow one to save an exponential amount of time.)

Another way to go is to apply compilation ideas from computational OT phonology to the (sets of) rules covering an extended projection. Karttunen (1998), extending ideas from Frank and Satta 1998 shows that in the finite-state approach to computational phonology it is possible to compute the OT competition offline. In this compilation step, a transducer is constructed that composes the violable constraints in a ("lenient") cascade. For the online application—i.e., when confronted with a specific

input structure—this transducer will determine the optimal candidate for the given underlying form, without effectively computing the candidate structures. The opposite processing direction works exactly the same way (not requiring any "backward generation", but nevertheless avoiding the ambiguity problem).

Exploiting the h-independence postulation of extended projections, an offline competition might also be incorporated in the LFG set-up: with restriction (17), the evaluation domain is already quite restricted; so one might go one step further and confine the allowable extended projections formally to partial tree structures that can be described by regular expressions (roughly, binary right-branching skeletons). If furthermore the OT constraints can talk about no more than a finite domain of f-structure "types" projected from these skeletons, then the relevant part of *Gen* plus the entire constraint system can be expressed by means of regular languages and regular relations. This means that one can effectively apply the idea of a lenient cascade of precompiled constraints also in syntax. Note that the language generated by the overall system is not restricted to the class of regular languages. The recursive structure of the input f-structure and the well-formedness principles applying on f-structure can enforce that the string language generated is contained in a higher class.

As with OT phonology, the resulting compiled grammar can be applied in both directions, using standard techniques; no special „backward generation" step is required. The overall complexity is reduced to the complexity of processing a single input with a classical (not comparison-based) grammar (solution to problem A.). One and the same compiled grammar can be used in comprehension and production—nevertheless ambiguous surface forms receive multiple analyses (solution to B.). Finally, the compiled cascade of constraints constitutes an implementation of the conception of the "inventory view" that Legendre et al. (1996) adopt. LF-unfaithful winners will play a role only locally within extended projections and only during compilation (preliminaries for a solution to C.).

6. Conclusion

I proposed the central parts of a formalization of Bresnan's (1998) OT-LFG model, arguing for some formal restrictions. The restrictions required to guarantee decidability seem to be quite plausible from the linguistic point of view.

In a model with global candidate generation, the parsing task requires an extra backward generation step. The resulting system will avoid the ambiguity problem, but the construction of global candidate analyses goes along with an immense complexity. The relative locality of constraint interaction can be exploited in a model interpreting constraint violability always local to the domain of an extended projection.

Based on this restriction one could even devise a model that compiles the entire OT competition offline, avoiding the complexity problem. Such a model would at the same time provide a very elegant manifestation of the "inventory view" on OT competition.

References

Bresnan, Joan. 1998. Optimal Syntax. To appear in *Optimality Theory: Phonology, Syntax, and Acquisition*, ed. Joost Dekkers, Frank van der Leeuw, and Jeroen van de Weijer. Oxford University Press.

Bresnan, Joan. 1999. Lexical-Functional Syntax. Ms. Stanford University.

Frank, Anette, Tracy H. King, Jonas Kuhn, and John Maxwell. 1998. Optimality Theory Style Constraint Ranking in Large-scale LFG Grammars. In *Proceedings of the Third LFG Conference*, ed. Miriam Butt and Tracy H. King. CSLI Proceedings Online.

Frank, Robert and Giorgio Satta. 1998. Optimality theory and the generative complexity of constraint violation. In *Computational Linguistics* 24:307–316.

Grimshaw, Jane. 1991. Extended Projection. Ms., Brandeis University.

Grimshaw, Jane. 1997. Projection, Heads, and Optimality. In *Linguistic Inquiry* 28: 373–422.

Grimshaw, Jane. 1998. Constraints on Constraints in Optimality Theoretic Syntax. Ms., Rutgers University.

Hale, Mark and Charles Reiss. 1998. Formal and Empirical Arguments concerning Phonological Acquisition. In *Linguistic Inquiry* 29:656–683.

Johnson, Mark. 1998. Optimality-theoretic Lexical Functional Grammar. To appear in *Proceedings of the 11th Annual CUNY Conference on Human Sentence Processing, Rutgers University, 1998.*

Karttunen, Lauri. 1998. The Proper Treatment of Optimality in Computational Phonology. In *Proceedings of the International Workshop on Finite-State Methods in Natural Language Processing, FSMNLP'98*, 1–12.

Kuhn, Jonas. 1999. Generation and Parsing in Optimality Theoretic Syntax—Issues in the Formalization of OT-LFG. Ms. To appear in CSLI volume on OT-LFG, ed. Peter Sells.

Legendre, Geraldine, Paul Smolensky, and Colin Wilson. 1996. When is less more? Faithfulness and minimal links in wh-chains. Ms. John-Hopkins University.

Smolensky, Paul. 1996. On the comprehension/production dilemma in child language. In *Linguistic Inquiry* 27:720–731.

Tesar, Bruce. 1995. Computational Optimality Theory. Doctoral Dissertation, University of Colorado.

Wedekind, Jürgen. 1999. Semantic-driven Generation with LFG- and PATR-style Grammars. In *Computational Linguistics* 25:277–281.

Institut für maschinelle Sprachverarbeitung
Universität Stuttgart
Azenbergstr. 12
70174 Stuttgart, Germany

jonas@ims.uni-stuttgart.de