Proceedings of the 40th Annual Meeting of the Association for
Computational Linguistics (ACL), Philadelphia, July 2002, pp. 48-55.

OT Syntax: Decidability of Generation-based Optimization[*]

Jonas Kuhn
Department of Linguistics
Stanford University
jonask@stanford.edu

Abstract

In Optimality-Theoretic Syntax, optimization with unrestricted expressive power on the side of the OT constraints is undecidable. This paper provides a proof for the decidability of optimization based on constraints expressed with reference to local subtrees (which is in the spirit of OT theory). The proof builds on Kaplan and Wedekind's (2000) construction showing that LFG generation produces context-free languages.

1 Introduction

Optimality-Theoretic (OT) grammar systems are an interesting alternative to classical formal grammars, as they construe the task of learning from data in a meaning-based way: a form is defined as grammatical if it is optimal (most harmonic) within a set of generation alternatives for an underlying logical form. The harmony of a candidate analysis depends on a language-specific ranking ($\gg$) of violable constraints, thus the learning task amounts to adjusting the ranking over a given set of constraints.

(1) Candidate A_1 is more harmonic than A_2 iff it incurs fewer violations of the highest-ranking constraint $C^i \in \mathcal{C}$ in which A_1 and A_2 differ.

The comparison-based setup of OT learning is closely related to discriminative learning approaches in probabilistic parsing (Johnson et al., 1999; Riezler et al., 2000; Riezler et al., 2002),[1] however the comparison of generation alternatives – rather than parsing alternatives – adds the possibility of systematically learning the basic language-specific grammatical principles (which in probabilistic parsing are typically fixed a priori, using either a treebank-derived or a manually written grammar for the given language). The "base grammar" assumed as given can be highly unrestricted in the OT setup. Using a linguistically motivated set of constraints, learning proceeds with a bias for *unmarked* linguistic structures (cf. e.g., (Bresnan et al., 2001)).

For computational OT syntax, an interleaving of candidate generation and constraint checking has been proposed (Kuhn, 2000). But the decidability of the optimization task in OT syntax, i.e., the identification of the optimal candidate(s) in a potentially infinite candidate set, has not been proven yet.[2]

2 Undecidability for unrestricted OT

Assume that the candidate set is characterized by a context-free grammar (cfg) G_1, plus one additional candidate '*yes*'. There are two constraints ($C^1 \gg C^2$): C^1 is violated if the candidate is neither '*yes*' nor a structure generated by a cfg G_2; C^2 is violated only by '*yes*'. Now, '*yes*' is in the language defined by this system iff there are no structures in G_1 that are also in G_2. But the emptiness problem for the intersection of two context-free languages is known to be undecidable, so the optimization task for unrestricted OT is undecidable too.[3]

However, it is not in the spirit of OT to have extremely powerful individual constraints; the explanatory power should rather arise from *interaction* of simple constraints.

3 OT-LFG

Following (Bresnan, 2000; Kuhn, 2000; Kuhn, 2001), we define a restricted OT system based on Lexical-Functional Grammar (LFG) representations: c(ategory) structure/f(unctional) structure

[*] This work was supported by a postdoctoral fellowship of the German Academic Exchange Service (DAAD).

[1] This is for instance pointed out by (Johnson, 1998).

[2] Most computational OT work so far focuses on candidates and constraints expressible as regular languages/rational relations, based on (Frank and Satta, 1998) (e.g., (Eisner, 1997; Karttunen, 1998; Gerdemann and van Noord, 2000)).

[3] Cf. also (Johnson, 1998) for the sketch of an undecidability argument and (Kuhn, 2001, 4.2, 6.3) for further constructions.

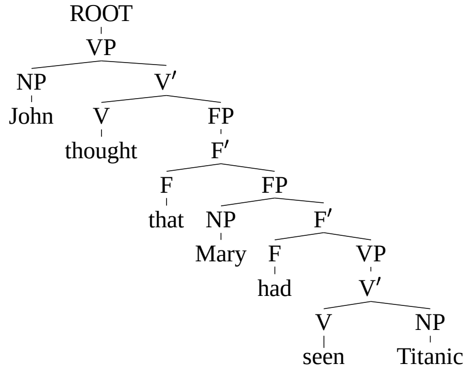
pairs $\langle T, \Phi\rangle$ like $\langle$(4),(5)$\rangle$. Each c-structure tree node is mapped to a node in the f-structure graph by the function ϕ. The mapping is specified by f-annotations in the grammar rules (below category symbols, cf. (2)) and lexicon entries (3).[4]

(2)
$$\begin{array}{lll}
\text{ROOT} & \to & \left\{ \begin{array}{c|c} \text{FP} & \text{VP} \\ \uparrow=\downarrow & \uparrow=\downarrow \end{array} \right\} \\
\text{FP} & \to & \left\{ \begin{array}{cc} \text{NP} & \text{FP} \\ (\uparrow\text{TOPIC})=\downarrow & \uparrow=\downarrow \\ (\uparrow\text{COMP* OBJ})=\downarrow & \end{array} \right. \\
 & & \left| \begin{array}{cc} \text{(NP)} & \text{F}' \\ (\uparrow\text{SUBJ})=\downarrow & \uparrow=\downarrow \end{array} \right\} \\
\text{F}' & \to & \begin{array}{c} \text{F} \\ \uparrow=\downarrow \end{array} \quad \left\{ \begin{array}{c|c} \text{FP} & \text{VP} \\ \uparrow=\downarrow & \uparrow=\downarrow \end{array} \right\} \\
\text{VP} & \to & \begin{array}{c} \text{(NP)} \\ (\uparrow\text{SUBJ})=\downarrow \end{array} \quad \begin{array}{c} \text{V}' \\ \uparrow=\downarrow \end{array} \\
\text{V}' & \to & \begin{array}{c} \text{V} \\ \uparrow=\downarrow \end{array} \quad \left\{ \begin{array}{c|c} \text{NP} & \text{FP} \\ (\uparrow\text{OBJ})=\downarrow & (\uparrow\text{COMP})=\downarrow \end{array} \right\}
\end{array}$$

(3)

Mary	NP	($\uparrow$PRED)=‘Mary’ ($\uparrow$NUM)=SG
that	F	
had	F	($\uparrow$TNS)=PAST
seen	V	($\uparrow$PRED)=‘see$\langle$ ($\uparrow$SUBJ) ($\uparrow$OBJ) $\rangle$’ ($\uparrow$ASP)=PERF
thought	V	($\uparrow$PRED)=‘think$\langle$ ($\uparrow$SUBJ) ($\uparrow$COMP) $\rangle$’ ($\uparrow$TNS)=PAST
laughed	V	($\uparrow$PRED)=‘laugh$\langle$ ($\uparrow$SUBJ) $\rangle$’ ($\uparrow$TNS)=PAST

(4) c-structure



(5) f-structure

$$\left[\begin{array}{ll}
\text{PRED} & \text{‘think}\langle(\uparrow\text{SUBJ})\ (\uparrow\text{COMP})\rangle\text{’} \\
\text{TNS} & \text{PAST} \\
\text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{‘John’} \\ \text{NUM} & \text{SG} \end{array}\right] \\
\text{COMP} & \left[\begin{array}{ll}
\text{PRED} & \text{‘see}\langle(\uparrow\text{SUBJ})\ (\uparrow\text{OBJ})\rangle\text{’} \\
\text{TNS} & \text{PAST} \\
\text{ASP} & \text{PERF} \\
\text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{‘Mary’} \\ \text{NUM} & \text{SG} \end{array}\right] \\
\text{OBJ} & \left[\begin{array}{ll} \text{PRED} & \text{‘Titanic’} \\ \text{NUM} & \text{SG} \end{array}\right]
\end{array}\right]
\end{array}\right]$$

The correct f-structure for a sentence is the minimal model satisfying all properly instantiated f-annotations.

In OT-LFG, the universe of possible candidates is defined by an LFG G_{inviol} (encoding inviolable principles, like an X-bar scheme). A particular candidate set is the set $Gen_{G_{inviol}}(\Phi_{in})$ – i.e., the c-/f-structure pairs in G_{inviol}, which have the input Φ_{in} as their f-structure. Constraints are expressed as local configurations in the c-/f-structure pairs. They have one of the following implicational forms:[5]

(6)
$$\begin{array}{ccc} N & \Rightarrow & N' \\ S & & S' \end{array}$$

where N, N' are descriptions of nonterminals of G_{inviol}; S, S' are standard LFG f-annotations of constraining equations with $\uparrow$ as the only f-structure metavariable.

(7)
$$\begin{array}{c} N \\ \hline \begin{array}{ccc} \rho & M & \sigma \\ & S & \end{array} \end{array} \quad \Rightarrow \quad \begin{array}{c} N' \\ \hline \begin{array}{ccc} \rho' & M' & \sigma' \\ & S' & \end{array} \end{array}$$

where N, N', M, M' are descriptions of nonterminals of G_{inviol}; N, N' refer to the mother in a local subtree configuration, M, M' refer to the same daughter category; $\rho, \rho', \sigma, \sigma'$ are regular expressions over nonterminals; S, S' are standard f-annotations as in (6).

Any of the descriptions can be maximally unspecific; (6) can for example be instantiated by the OPSPEC constraint ($\uparrow$OP)=+ $\Rightarrow$ (DF $\uparrow$) (an operator must be the value of a discourse function, (Bresnan, 2000)) with the category information unspecified.

An OT-LFG system $\mathcal{O}$ is thus characterized by a base grammar and a set of constraints, with a language-specific ranking relation $\gg_{\mathcal{L}}$:
$\mathcal{O} = \langle G_{inviol}, \langle \mathcal{C}, \gg_{\mathcal{L}}\rangle\rangle$.
The evaluation function $Eval_{\langle \mathcal{C}, \gg_{\mathcal{L}}\rangle}$ picks the most harmonic from a set of candidates, based on the constraints and ranking. The language (set of analyses)[6] generated by an OT system is defined as

$$\begin{array}{l} L(\mathcal{O}) = \{\langle T_j, \Phi_j\rangle \in G_{inviol} \\ \qquad | \, \exists \Phi_{in} : \langle T_j, \Phi_j\rangle \in Eval_{\langle \mathcal{C}, \gg_{\mathcal{L}}\rangle}(Gen_{G_{inviol}}(\Phi_{in}))\} \end{array}$$

4 LFG generation

Our decidability proof for generation-based optimization builds on the result of (Kaplan and Wedekind, 2000) (K&W00) that LFG generation produces context-free languages.

[4] $\downarrow$ abbreviates $\phi(*)$, i.e., the present category's ϕ image; $\uparrow$ abbreviates $\phi(\mathcal{M}*)$, i.e., the f-structure corresponding to the present node's mother category.

[5] Note that with GPSG-style category-level feature percolation it is possible to refer to (finitely many) nonlocal configurations at the local tree level.

[6] The string language is obtained by taking the terminal string of the c-structure part of the analyses.

(8) Given an arbitrary LFG grammar G and a cycle-free f-structure Φ, a cfg G' can be constructed that generates exactly the strings to which G assigns the f-structure Φ.

I will refer to the resulting cfg G' as $KW(G, \Phi)$. K&W00 present a constructive proof, folding all f-structural contributions of lexical entries and LFG rules into the c-structural rewrite rules (which is possible since we know in advance the range of f-structural objects that can instantiate the f-structure meta-variables in the rules). I illustrate the specialization steps with grammar (2) and lexicon (3) and for generation from f-structure (5).

Initially, the generalized format of right-hand sides in LFG rules is converted to the standard context-free notation (resolving regular expressions by explicit disjunction or recursive rules). F-structure (5) contains five substructures: the root f-structure, plus the embedded f-structures under the paths SUBJ, COMP, COMP SUBJ, and COMP OBJ. Any relevant metavariable (↑, ↓) in the grammar must end up instantiated to one of these. So for each path from the root f-structure, a distinct variable is introduced: v, subscripted with the (abbreviated and possibly empty) feature path: $v, v_s, v_c, v_{cs}, v_{co}$.

Rule augmentation step 1 adds to each category name a concrete f-structure to which the category corresponds. So for FP, we get FP:v, FP:v_s, FP:v_c, FP:v_{cs}, and FP:v_{co}. The rules are multiplied out to cover all combinations of augmented categories obeying the original f-annotations.[7] Step 2 adds a set of instantiated f-annotation schemes to each symbol, based on the instantiation of metavariables from step 1. One instance of the lexicon entry *Mary* look as follows:

(9) $$\text{NP:}v_{cs}\text{:}\left\{\begin{array}{c}(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_{cs}\ \text{NUM})\text{=SG}\end{array}\right\} \quad\rightarrow\quad \text{Mary}$$

The rules are again multiplied out to cover all combinations for which the set of f-constraints on the mother is the union of all daughters' f-constraints, plus the appropriately instantiated rule-specific annotations. So, for the VP rule based on the categories $\text{NP:}v_{cs}\text{:}\left\{\begin{array}{c}(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_{cs}\ \text{NUM})\text{=SG}\end{array}\right\}$ and $\text{V}'\text{:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`laugh'}\\(v_c\ \text{TNS})\text{=PAST}\\v_c = v_c\end{array}\right\}$, we get the rule

$$\text{VP:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{SUBJ}) = v_{cs}\\(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_{cs}\ \text{NUM})\text{=SG}\\(v_c\ \text{PRED})\text{=`laugh'}\\(v_c\ \text{TNS})\text{=PAST}\\v_c = v_c\end{array}\right\}\rightarrow$$
$$\text{NP:}v_{cs}\text{:}\left\{\begin{array}{c}(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_{cs}\ \text{NUM})\text{=SG}\end{array}\right\}\text{V}'\text{:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`laugh'}\\(v_c\ \text{TNS})\text{=PAST}\\v_c = v_c\end{array}\right\}$$

With this bottom-up construction it is ensured that each new category ROOT:v:$\{\ldots\}$ (corresponding to the original root symbol) contains a complete possible collection of instantiated f-constraints. To exclude analyses whose f-structure is *not* Φ (for which we are generating strings) a new start symbol is introduced "above" the original root symbol. Only for the sets of f-constraints that have Φ as their minimal model, rules of the form ROOT′ → ROOT:v:$\{\ldots\}$ are introduced (this also excludes inconsistent f-constraint sets).

With the cfg $KW(G, \Phi)$, standard techniques for cfg's can be applied, e.g., if there are infinitely many possible analyses for a given f-structure, the smallest one(s) can be produced, based on the pumping lemma for context-free languages. Grammar (2) does indeed produce infinitely many analyses for the input f-structure (5). It overgenerates in several respects: The functional projection FP can be stacked due to recursions like the following (with the augmented FP reoccuring in the F′ rules):

$$\text{FP:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`see}\langle\ \ldots\rangle\text{'}\\(v_c\ \text{TNS})\text{=PAST}\\(v_c\ \text{SUBJ}) = v_{cs}\\(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_c\ \text{OBJ}) = v_{co}\\(v_{co}\ \text{PRED})\text{=`Titanic'}\\v_c = v_c\end{array}\right\}\rightarrow\text{F}'\text{:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`see}\langle\ \ldots\rangle\text{'}\\(v_c\ \text{TNS})\text{=PAST}\\(v_c\ \text{SUBJ}) = v_{cs}\\(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_c\ \text{OBJ}) = v_{co}\\(v_{co}\ \text{PRED})\text{=`Titanic'}\\v_c = v_c\end{array}\right\}$$

$$\text{F}'\text{:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`see}\langle\ \ldots\rangle\text{'}\\(v_c\ \text{TNS})\text{=PAST}\\(v_c\ \text{SUBJ}) = v_{cs}\\(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_c\ \text{OBJ}) = v_{co}\\(v_{co}\ \text{PRED})\text{=`Titanic'}\\v_c = v_c\end{array}\right\}\rightarrow\text{F:}v_c\text{:}\emptyset\ \text{FP:}v_c\text{:}\left\{\begin{array}{c}(v_c\ \text{PRED})\text{=`see}\langle\ \ldots\rangle\text{'}\\(v_c\ \text{TNS})\text{=PAST}\\(v_c\ \text{SUBJ}) = v_{cs}\\(v_{cs}\ \text{PRED})\text{=`Mary'}\\(v_c\ \text{OBJ}) = v_{co}\\(v_{co}\ \text{PRED})\text{=`Titanic'}\\v_c = v_c\end{array}\right\}$$

F:v_c:$\emptyset$ is one of the augmented categories we get for *that* in (3), so KW((2),(5)) generates an arbitrary number of *that*s on top of any FP. A similar repetition effect will arise for the auxiliary *had*.[8] Other choices in generation arise from the freedom of generating the subject in the specifier of VP or FP and from the possibility of (unbounded) topicalization of the object (the first disjunction of the FP rule in (2)

[7] VP:v_c → NP:v_{cs} V′:v_c is allowed, while VP:v_c → NP:v_{cs} V′:v_{co} is excluded, since the ↑=↓ annotation of V′ in the VP rule (2) enforces that $\phi(\text{VP}) = \phi(\text{V}')$.

[8] The F^0 entries do not contribute any PRED value, which would exclude doubling due to the instantiated symbol character of PRED values (cf. K&W00, fn. 2).

contains a functional-uncertainty equation):

(10) a. John thought that Titanic, Mary had seen.
b. Titanic, John thought that Mary had seen.

5 LFG generation in OT-LFG

While grammar (2) would be considered defective as a classical LFG grammar, it constitutes a reasonable example of a candidate generation grammar (G_{inviol}) in OT. Here, it is the OT constraints that enforce language-specific restrictions, so G_{inviol} has to ensure that all candidates are generated in the first place. For instance, expletive elements as *do* in *Who do you know* will arise by passing a recursion in the cfg constructed during generation. A candidate containing such a vacuous cycle can still become the winner of the OT competition if the Faithfulness constraint punishing expletives is outranked by some constraint favoring an aspect of the recursive structure. So the harmony is increased by going through the recursion a certain number of times. It is for this very reason, that *Who do you know* is predicted to be grammatical in English.

So, in OT-LFG it is not sufficient to apply just the KW construction; I use an additional step: prior to application of KW, the LFG grammar G_{inviol} is converted to a different form $O_{\mathcal{C}}(G_{inviol})$ (depending on the constraint set $\mathcal{C}$), which is still an LFG grammar but has category symbols which reflect local constraint violations. When the KW construction is applied to $O_{\mathcal{C}}(G_{inviol})$, all "pumping" structures generated by the cfg $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$ can indeed be ignored since all OT-relevant candidates are already contained in the finite set of nonrecursive structures. So, finally the ranking of the constraints is taken into consideration in order to determine the harmony of the candidates in this finite subset.

6 The conversion $O_{\mathcal{C}}(G_{inviol})$

Preprocessing Like K&W00, I assume an initial conversion of the c-structure part of rules into standard context-free form, i.e., the right-hand side is a category string rather than a regular expression. This ensures that for a given local subtree, each constraint (of form (6) or (7)) can be applied only a finite number of times: if l is the arity of the longest right-hand side of a rule, the maximal number of local violations is l (since some constraints of type (7) can be instantiated to all daughters).

Grammar conversion With the number of local violations bounded, we can encode all candidate distinctions with respect to constraint violations at the local-subtree level with finite means: The set of categories in the newly constructed LFG grammar $O_{\mathcal{C}}(G_{inviol})$ is the finite set

(11) $\mathcal{N}_{O_{\mathcal{C}}(G_{inviol})}$: *the set of categories in* $O_{\mathcal{C}}(G_{inviol})$

$\{N{:}\langle n^1, n^2, n^3 \ldots n^k\rangle \mid$
N a nonterminal symbol of G_{inviol},
k the size of the constraint set $\mathcal{C}$,
$0 \leq n^i \leq l$,
l the arity of the longest rhs in rules of $G_{inviol}\}$

The rules in $O_{\mathcal{C}}(G_{inviol})$ are constructed in such a way that for each rule

$$\begin{array}{l} \mathrm{X}_0 \rightarrow \mathrm{X}_1 \ldots \mathrm{X}_m \\ \quad\;\; T_1 \quad\; T_m \end{array}$$

in G_{inviol} and each sequence $\langle n_0^1, n_0^2 \ldots n_0^k\rangle$, $0 \leq n_0^i \leq l$, all rules of the form

$$\begin{array}{l} \mathrm{X}_0{:}\langle n_0^1, n_0^2 \ldots n_0^k\rangle \rightarrow \underset{T'_1}{\mathrm{X}_1{:}\langle n_1^1 \ldots n_1^k\rangle} \ldots \underset{T'_m}{\mathrm{X}_m{:}\langle n_m^1 \ldots n_m^k\rangle}, \\ 0 \leq n_j^i \leq l \end{array}$$

are included such that n_0^i (the number of violations of constraint C^i incurred local to the rule) and the f-annotations $T'_1 \ldots T'_m$ are specified as follows:

(12) for C^i of form (6) $\begin{bmatrix} N & \Rightarrow & N' \\ S & & S' \end{bmatrix}$:

a. $n_0^i = 0$; $T'_j = T_j$ $(1 \leq j \leq m)$
if X_0 does not match the condition N;

b. $n_0^i = 0$; $T'_1 = T_1 \wedge \neg S$; $T'_j = T_j$ $(2 \leq j \leq m)$
if X_0 matches N;

c. $n_0^i = 0$; $T'_1 = T_1 \wedge S \wedge S'$; $T'_j = T_j$ $(2 \leq j \leq m)$
if X_0 matches both N and N';

d. $n_0^i = 1$; $T'_1 = T_1 \wedge S$; $T'_j = T_j$ $(2 \leq j \leq m)$
if X_0 matches N but not N';

e. $n_0^i = 1$; $T'_1 = T_1 \wedge S \wedge \neg S'$; $T'_j = T_j$ $(2 \leq j \leq m)$
if X_0 matches both N and N';

(13) for C^i of form (7) $\left[\begin{array}{ccc} N & \Rightarrow & N' \\ \rho\ M\ \sigma & & \rho'\ M'\ \sigma' \\ S & & S' \end{array}\right]$:

a. $n_0^i = 0$; $T'_j = T_j$ $(1 \leq j \leq m)$
if X_0 does not match the condition N;

b. $n_0^i = \sum_{j=1}^{m} d_j$; $T'_j = \delta(T_j, S, S')$ $(1 \leq j \leq m)$,
where

i. $d_j = 0$; $\delta(T_j, S, S') = T_j$
if X_j does not match M, or $\mathrm{X}_1 \ldots \mathrm{X}_{j-1}$ do not match ρ, or $\mathrm{X}_{j+1} \ldots \mathrm{X}_m$ do not match σ;

ii. $d_j = 0$; $\delta(T_j, S, S') = T_j \wedge S \wedge S'$
if X_0 matches both N and N'; X_j matches both M and M'; $\mathrm{X}_1 \ldots \mathrm{X}_{j-1}$ match ρ and ρ'; $\mathrm{X}_{j+1} \ldots \mathrm{X}_m$ match σ and σ';

iii. $d_j = 0$; $\delta(T_j, S, S') = T_j \wedge \neg S$
if X_0 matches both N and N'; X_j matches both M and M'; $\mathrm{X}_1 \ldots \mathrm{X}_{j-1}$ match ρ and ρ'; $\mathrm{X}_{j+1} \ldots \mathrm{X}_m$ match σ and σ';

iv. $d_j = 1$; $\delta(T_j, S, S') = T_j \wedge S$
if X_0 matches N, X_j matches M, $\mathrm{X}_1 \ldots \mathrm{X}_{j-1}$ match ρ, $\mathrm{X}_{j+1} \ldots \mathrm{X}_m$ match σ, but (at least) one of them does not match the respective description in the consequent (N', M', ρ', σ');

v. $d_j = 1$; $\delta(T_j, S, S') = T_j \wedge S \wedge \neg S'$
if X_0 matches both N and N'; X_j matches both M and M'; $\mathrm{X}_1 \ldots \mathrm{X}_{j-1}$ match ρ and ρ'; $\mathrm{X}_{j+1} \ldots \mathrm{X}_m$ match σ and σ'.

Note that the constraint profile of the daughter categories does not play any role in the determination of constraint violations local to the subtree under consideration (only the sequences n_0^i are restricted by the conditions (12) and (13)). So for each new rule type, all combinations of constraint profiles on the daughters are constructed (creating a large but finite number of rules).[9] This ensures that no sentence that can be parsed (or generated) by G_{inviol} is excluded from $O_\mathcal{C}(G_{inviol})$ (as stated by fact (14)):[10]

(14) *Coverage preseveration*
All strings generated by an LFG grammar G are also generated by $O_\mathcal{C}(G)$.

The original G analysis can be recovered from an $O_\mathcal{C}(G)$ analysis by applying a projection function *Cat* to all c-structure categories:

Cat$(N{:}\langle n^1, n^2 \ldots n^k \rangle) = N$
for every category in $\mathcal{N}_{O_\mathcal{C}(G_{inviol})}$ (11)

We can overload the function name *Cat* with a function applying to the set of analyses produced by an LFG grammar G by defining

Cat$(G) = \{\langle T, \Phi \rangle \mid \langle T', \Phi \rangle \in G$, T is derived from T' by applying *Cat* to all category symbols $\}$.

Coverage preservation of the $O_\mathcal{C}$ construction holds also for the projected c-category skeleton (cf. the argumentation in fn. 10):

(15) *C-structure level coverage preservation*
For an LFG grammar G: *Cat*$(O_\mathcal{C}(G)) = G$

Each category in $O_\mathcal{C}(G)$ encodes the number of local violations for all constraints. Since all constraints are locally evaluable by assumption, *all* constraints violated by a candidate analysis have to be incurred local to some subtree. Hence the total number of constraint violations incurred by a candidate can be computed by simply summing over all category-encoded local violation profiles:

(16) *Total number of constraint violations*
Let *Nodes*(T) be the multiset of categories occurring in the c-structure tree T, then the total number of violations of constraint C^i incurred by an analysis $\langle T, \Phi \rangle \in O_\mathcal{C}(G_{inviol})$ is

$$\#_{C^i}(T) = \sum\nolimits_{N:\langle \ldots n^i \ldots \rangle \in Nodes(T)} n^i$$

Define
Total$_\mathcal{C}(T) = \langle \#_{C^1}(T), \#_{C^2}(T), \ldots \#_{C^k}(T) \rangle$

7 Applying KW on $O_\mathcal{C}(G_{inviol})$

Since $O_\mathcal{C}(G_{inviol})$ is a standard LFG grammar, we can apply the KW construction to it to get a cfg for a given f-structure Φ_{in}. The category symbols then have the form X:$\langle n^1, \ldots, n^k \rangle{:}v{:}D$, with v and D arising from the KW construction. We can overload the projection function *Cat* again such that *Cat*$(u{:}v{:}w{:}x) = u$ for all augmented category symbol of the new format; likewise *Cat*(G) for G a cfg.

Since the $O_\mathcal{C}$ construction (strongly) preserves the language generated, coverage preservation holds also after the application of KW to $O_\mathcal{C}(G_{inviol})$ and G_{inviol}, respectively:

(17) *Cat*$(KW(O_\mathcal{C}(G_{inviol}), \Phi_{in})) =$
Cat$(KW(G_{inviol}, \Phi_{in}))$

But since the symbols in $O_\mathcal{C}(G_{inviol})$ reflect local constraint violations, *Cat*$(KW(O_\mathcal{C}(G_{inviol}), \Phi_{in}))$ has the property that *all instances of recursion* in the

[9] For one rule/constraint combination several new rules can result; e.g., if the right-hand side of a rule (X_0) matches both the antecedent (N) and the consequent (N') category description of a constraint of form (6), three clauses apply: (12b), (12c), and (12d). So, we get two new rules with the count of 0 local violations of the constraint and two rules with count 1, with a difference in the f-annotations.

[10] Providing all possible combinations of augmented category symbols on the right-hand rule sides in $O_\mathcal{C}(G)$ ensures that the newly constructed rules can be reached from the root symbol in a derivation. It is also guaranteed that whenever a rule R in G contributes to an analysis, at least one of the rules constructed from R will contribute to the corresponding analysis in $O_\mathcal{C}(G)$. This is ensured since the subclauses in (12) and (13) cover the full space of logical possibilities.

resulting cfg *create candidates that are at most as harmonic as their non-recursive counterparts.* Assuming a projection function $\mathit{CatCount}(u{:}v{:}w{:}x) = u{:}v$, we can state more formally:

(18) If T_1 and T_2 are *CatCount* projections of trees produced by the cfg $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$, using exactly the same rules, and T_2 contains a superset of the nodes that T_1 contains, then
$n_1^i \leq n_2^i$, for all n_1^i, n_2^i $(i = 1..k)$
from $\langle n_1^1 \ldots n_1^i \ldots n_1^k \rangle = \mathit{Total}_{\mathcal{C}}(T_1)$,
and $\langle n_2^1 \ldots n_2^i \ldots n_2^k \rangle = \mathit{Total}_{\mathcal{C}}(T_2)$.

This fact follows from definition of *Total* (16): the violation counts in the additional nodes in T_2 will add to the total of constraint violations (and if none of the additional nodes contains any local constraint violation at all, the total will be the same as in T_1). Intuitively, the effect of the augmentation of the category format is that certain recursions in the pure KW construction (which one may think of as a loop) are unfolded, leading to a longer loop. The new loop is sufficiently large to make all relevant distinctions.

This result can be directly exploited in processing: if all non-recursive analyses are generated (of which there are only finitely many) it is guaranteed that a subset of the optimal candidates is among them. If the grammar does not contain any *violation-free* recursion, we even know that we have generated all optimal candidates.

(19) A recursion with the derivation path $A \Rightarrow \ldots \Rightarrow A$ is called *violation-free* iff all categories dominated by the upper occurrence of A, but not dominated by the lower occurrence of A have the form $N : \langle n^1, n^2 \ldots n^k \rangle$ with $n^i = 0, i = 1..k$

Note that if there *is* an applicable violation-free recursion, the set of optimal candidates is infinite; so if the constraint set is set up properly in a linguistic analysis, one would assume that violation-free recursion should not arise. (Kuhn, 2000) excludes the application of such recursions by a similar condition as offline parsability (which excludes vacuous recursions over a string in parsing), but with the KW construction, this condition is not necessary for decidability of the generation-based optimization task. The cfg produced by KW can be transformed further to only generate the optimal candidates according to the constraint ranking $\gg_{\mathcal{L}}$ of the OT system $\mathcal{O} = \langle G_{inviol}, \langle \mathcal{C}, \gg_{\mathcal{L}} \rangle \rangle$, eliminating all but the violation-free recursions in the grammar:

(20) *Creating a cfg that produces all optimal candidates*
a. Define
$\mathcal{T}_{\Phi_{in}}^{NR} = \{T \in KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in}) \mid T$ contains no recursion $\}$.
$\mathcal{T}_{\Phi_{in}}^{NR}$ is finite and can be easily computed, by keeping track of the rules already used in an analysis.

b. Redefine $\mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}$ to apply on a set of context-free analyses with augmented category symbols with counts of local constraint violations:
$\mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}(\mathcal{T}) = \{T \in \mathcal{T} \mid T$ is maximally harmonic in $\mathcal{T}$, under ranking $\gg_{\mathcal{L}}\}$
Using the function *Total* defined in (16), this function is straightforward to compute for finite sets, i.e., in particular $\mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}(\mathcal{T}_{\Phi_{in}}^{NR})$.

c. Augment the category format further by one index component.[11] Introduce index $h = 0$ for all categories in $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$ of the form X:$\langle n^1, \ldots n^k \rangle{:}v{:}D$, where $n^i = 0$ for $i = 1..k$. Introduce a new unique index $h > 1$ for each node of the form X:$\langle n^1, \ldots n^k \rangle{:}v{:}D$, where $n^i \neq 0$ for some n^i $(1 \leq i \leq k)$ occurring in the analyses $\mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}(\mathcal{T}_{\Phi_{in}}^{NR})$ (i.e., different occurrences of the same category are distinguished).

d. Construct the cfg

$$G_{\Phi_{in}}^{OT} = \langle N_{\Phi_{in}}^{OT}, T_{\Phi_{in}}^{OT}, \mathrm{S}_{\Phi_{in}}^{OT}, R_{\Phi_{in}}^{OT} \rangle,$$

where $N_{\Phi_{in}}^{OT}, T_{\Phi_{in}}^{OT}$ are the indexed symbols of step c.; $\mathrm{S}_{\Phi_{in}}^{OT}$ is a new start symbol; the rules $R_{\Phi_{in}}^{OT}$ are (i) those rules from $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$ which were used in the analyses in $\mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}(\mathcal{T}_{\Phi_{in}}^{NR})$ – with the original symbols replaced by the indexed symbols –, (ii) the rules in $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$, in which the mother category and all daughter categories are of the form X:$\langle n^1, \ldots n^k \rangle{:}v{:}D$, $n^i = 0$ for $i = 1..k$ (with the new index 0 added), and (iii) one rule $\mathrm{S}_{\Phi_{in}}^{OT} \rightarrow \mathrm{S}_j{:}h$ for each of the indexed versions $\mathrm{S}_j{:}h$ of the start symbols of $KW(O_{\mathcal{C}}(G_{inviol}), \Phi_{in})$.

With the index introduced in step (20c), the original recursion in the cfg is eliminated in all but the violation-free cases. The grammar $\mathit{Cat}(G_{\Phi_{in}}^{OT})$ produces (the c-structure of) the set of optimal candidates for the input Φ_{in}:[12]

(21) $\mathit{Cat}(G_{\Phi_{in}}^{OT}) =$
$\{T \mid \langle T, \Phi_{in} \rangle \in \mathit{Eval}_{\langle \mathcal{C}, \gg_{\mathcal{L}} \rangle}(\mathit{Gen}_{G_{inviol}}(\Phi_{in}))\}$,
i.e., the set of c-structures for the optimal candidates for input f-structure Φ_{in} according to the OT system $\mathcal{O} = \langle G_{inviol}, \langle \mathcal{C}, \gg_{\mathcal{L}} \rangle \rangle$.

[11] The projection function *Cat* is again overloaded to also remove the index on the categories.

[12] Like K&W00, I make the assumption that the input f-structure in generation is fully specified (i.e., all the candidates have the form $\langle T, \Phi_{in} \rangle$), but the result can be extended to allow for the addition of a finite amount of f-structure information in generation. Then, the specified routine is computed separately for each possible f-structural extension and the results are compared in the end.

8 Proof

To prove fact (21) we will show that the c-structure of an arbitrary candidate analysis generated from Φ_{in} with G_{inviol} is contained in $\mathit{Cat}(G^{OT}_{\Phi_{in}})$ iff all other candidates are equally or less harmonic.

Take an arbitrary candidate c-structure T generated from Φ_{in} with G_{inviol} such that $T \in \mathit{Cat}(G^{OT}_{\Phi_{in}})$. We have to show that all other candidates T' generated from Φ_{in} are equally or less harmonic than T. Assume there were a T' that is more harmonic than T. Then there must be some constraint $C^i \in \mathcal{C}$, such that T' violates C^i fewer times than T does, and C^i is ranked higher than any other constraint in which T and T' differ. Constraints have to be incurred within some local subtree; so T must contain a local violation configuration that T' does not contain, and by the construction (12)/(13) the $O_{\mathcal{C}}$-augmented analysis of T – call it $O_{\mathcal{C}}(T)$ – must make use of some violation-marked rule not used in $O_{\mathcal{C}}(T')$. Now there are three possibilities:
(i) Both $O_{\mathcal{C}}(T)$ and $O_{\mathcal{C}}(T')$ are free of recursion. Then the fact that $O_{\mathcal{C}}(T')$ avoids the highest-ranking constraint violation excludes T from $\mathit{Cat}(G^{OT}_{\Phi_{in}})$ (by construction step (20b)). This gives us a contradiction with our assumption.
(ii) $O_{\mathcal{C}}(T)$ contains a recursion and $O_{\mathcal{C}}(T')$ is free of recursion. If the recursion in $O_{\mathcal{C}}(T)$ is violation-free, then there is an equally harmonic recursion-free candidate T''. But this T'' is also less harmonic than $O_{\mathcal{C}}(T')$, such that it would have been excluded from $\mathit{Cat}(G^{OT}_{\Phi_{in}})$ too. This again means that $O_{\mathcal{C}}(T)$ would also be excluded (for lack of the relevant rules in the non-recursive part). On the other hand, if it were the recursion in $O_{\mathcal{C}}(T)$ that incurred the additional violation (as compared to $O_{\mathcal{C}}(T')$), then there would be a more harmonic recursion-free candidate T'''. However, this T''' would exclude the presence of $O_{\mathcal{C}}(T)$ in $G^{OT}_{\Phi_{in}}$ by construction step (20c,d) (only violation-free recursion is possible). So we get another contradiction to the assumption that $T \in \mathit{Cat}(G^{OT}_{\Phi_{in}})$.
(iii) $O_{\mathcal{C}}(T')$ contains a recursion. If this recursion is violation-free, we can pick the equally harmonic candidate avoiding the recursion to be our $O_{\mathcal{C}}(T')$, and we are back to case (i) and (ii). Likewise, if the recursion in $O_{\mathcal{C}}(T')$ does incur some violation, not using the recursion leads to an even more harmonic candidate, for which again cases (i) and (ii) will apply.
All possible cases lead to a contradiction with the assumptions, so no candidate is more harmonic than our $T \in \mathit{Cat}(G^{OT}_{\Phi_{in}})$.

We still have to prove that if the c-structure T of a candidate analysis generated from Φ_{in} with G_{inviol} is equally or more harmonic than all other candidates, then it is contained in $\mathit{Cat}(G^{OT}_{\Phi_{in}})$. We can construct an augmented version T' of T, such that $\mathit{Cat}(T') = T$ and then show that there is a homomorphism mapping T' to some analysis $T'' \in G^{OT}_{\Phi_{in}}$ with $\mathit{Cat}(T'') = T$.
We can use the constraint marking construction $O_{\mathcal{C}}$ and the KW construction to construct the tree T' with augmented category symbols of the analysis T. The result of K&W00 plus (17) guarantee that $\mathit{Cat}(T') = T$. Now, there has to be a homomorphism from the categories in T' to the categories of some analysis in $G^{OT}_{\Phi_{in}}$. $G^{OT}_{\Phi_{in}}$ is also based on $KW(G_{inviol}, \Phi_{in})$ (with an additional index h on each category and some categories and rules of $KW(G_{inviol}, \Phi_{in})$ having no counterpart in $G^{OT}_{\Phi_{in}}$).
Since we know that T is equally or more harmonic than any other candidate generated from Φ_{in}, we know that the augmented tree T' either contains no recursion or only violation-free recursion. If it does contain such violation-free recursions we map all categories N on the recursion paths to the indexed form N:0, and furthermore consider the variant of T' avoiding the recursion(s). For our (non-recursive) tree, there is guaranteed to be a counterpart in the finite set of non-recursive trees in $G^{OT}_{\Phi_{in}}$ with all categories pairwise identical apart from the index h in $G^{OT}_{\Phi_{in}}$. We pick this tree and map each of the categories in T' to the h-indexed counterpart. The existence of this homomorphism guarantees that an analysis $T'' \in G^{OT}_{\Phi_{in}}$ exists with $\mathit{Cat}(T'') = \mathit{Cat}(T') = T$. QED

9 Conclusion

We showed that for OT-LFG systems in which all constraints can be expressed relative to a local subtree in c-structure, the generation task from (noncyclic[13]) f-structures is solvable. The infinity of

[13] The non-cyclicity condition is inherited from K&W00; in linguistically motivated applications of the LFG formalism, cru-

the conceptually underlying candidate set does not preclude a computational approach. It is obvious that the construction proposed here has the purpose of bringing out the principled computability, rather than suggesting a particular algorithm for implementation. However on this basis, an implementation can be easily devised.

The locality condition on constraint-checking seems unproblematic for linguistically relevant constraints, since a GPSG-style slash mechanism permits reference to (finitely many) nonlocal configurations from any given category (cf. fn. 5).[14]

Decidability of generation-based optimization (from a given input f-structure) alone does not imply that the recognition and parsing tasks for an OT grammar system defined as in sec. 3 are decidable: for these tasks, a string is given and it has to be shown that the string is optimal for *some* underlying input f-structure (cf. (Johnson, 1998)). However, a similar construction as the one presented here can be devised for parsing-based optimization (even for an LFG-style grammar that does not obey the offline parsability condition). So, if the language generated by an OT system is defined based on (strong) bidirectional optimality (Kuhn, 2001, ch. 5), decidability of both the general parsing and generation problem follows.[15] For the unidirectionally defined OT language (as in sec. 3), decidability of parsing can be guaranteed under the assumption of a contextual recoverability condition in parsing (Kuhn, in preparation).

References

Joan Bresnan, Shipra Dingare, and Christopher Manning. 2001. Soft constraints mirror hard constraints: Voice and person in English and Lummi. In *Proceedings of the LFG 2001 Conference*. CSLI Publications.

Joan Bresnan. 2000. Optimal syntax. In Joost Dekkers, Frank van der Leeuw, and Jeroen van de Weijer, editors, *Optimality Theory: Phonology, Syntax, and Acquisition*. Oxford University Press.

Jason Eisner. 1997. Efficient generation in primitive optimality theory. In *Proceedings of the ACL 1997*, Madrid.

Robert Frank and Giorgio Satta. 1998. Optimality theory and the generative complexity of constraint violation. *Computational Linguistics*, 24(2):307–316.

Dale Gerdemann and Gertjan van Noord. 2000. Approximation and exactness in finite state Optimality Theory. In *SIGPHON 2000, Finite State Phonology. 5th Workshop of the ACL Special Interest Group in Comp. Phonology*, Luxembourg.

Mark Johnson, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic “unification-based” grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL’99), College Park, MD*, pages 535–541.

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

Ronald M. Kaplan and Jürgen Wedekind. 2000. LFG generation produces context-free languages. In *Proceedings of COLING-2000*, pages 297–302, Saarbrücken.

Lauri Karttunen. 1998. The proper treatment of optimality in computational phonology. In *Proceedings of the Internat. Workshop on Finite-State Methods in Natural Language Processing, FSMNLP’98*, pages 1–12.

Jonas Kuhn. 2000. Processing Optimality-theoretic syntax by interleaved chart parsing and generation. In *Proceedings of ACL 2000*, pages 360–367, Hongkong.

Jonas Kuhn. 2001. *Formal and Computational Aspects of Optimality-theoretic Syntax*. Ph.D. thesis, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart.

Jonas Kuhn. in preparation. Decidability of generation and parsing for OT syntax. Ms., Stanford University.

Stefan Riezler, Detlef Prescher, Jonas Kuhn, and Mark Johnson. 2000. Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and EM training. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL’00), Hong Kong*, pages 480–487.

Stefan Riezler, Dick Crouch, Ron Kaplan, Tracy King, John Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. This conference.

cial use of cyclicity in underlying semantic feature graphs has never been made.

[14] A hypothetical constraint that is excluded would be a parallelism constraint comparing two subtree structures of arbitrary depth. Such a constraint seems unnatural in a model of grammaticality. Parallelism of conjuncts *does* play a role in models of human parsing preferences; however, here it seems reasonable to assume an upper bound on the depth of parallel structures to be compared (due to memory restrictions).

[15] Parsing: for a given string, parsing-based optimization is used to determine the optimal underlying f-structure; then generation-based optimization is used to check whether the original string comes out optimal in this direction too. Generation is symmetrical, starting with an f-structure.