A UNIFICATION–BASED ID/LP PARSING SCHEMA

Frank Morawietz*
Universität Tübingen
Wilhelmstr. 113
72074 Tübingen
Germany
frank@sfs.nphil.uni-tuebingen.de

Abstract

In contemporary natural language formalisms like HPSG (Pollard and Sag 1994) the ID/LP format is used to separate the information on dominance from the one on linear precedence thereby allowing significant generalizations on word order. In this paper, we define unification ID/LP grammars. But as mentioned in Seiffert (1991) there are problems concerning the locality of the information determining LP acceptability during parsing. Since one is dealing with partially specified data, the information that is relevant to decide whether the local tree under construction is LP acceptable might be instantiated further during processing. In this paper we propose a modification of the Earley/Shieber algorithm on direct parsing of ID/LP grammars. We extend the items involved to include the relevant underspecified information using it in the completion steps to ensure the acceptability of the resulting structure. Following Sikkel (1993) we define it not as an algorithm, but as a parsing schema to allow the most abstract representation.

1 Introduction

The immediate dominance/linear precedence distinction was introduced into linguistic formalisms to encode word order generalizations by Gazdar, Klein, Pullum and Sag (1985) for GPSG and contemporary formalisms like HPSG (Pollard and Sag 1994) still want to express linearization facts with LP rules. But HPSG does not provide definitions for how to incorporate the ID/LP format into the formalism. This paper tries to handle the ID/LP distinction on another level. Instead of incorporating it into the theory, the information will be used during processing to determine validity of structures.

The simplest approach to parsing of ID/LP grammars is to fully expand the grammar into the underlying phrase structure grammar. But the expansion creates a huge number of grammar rules[1] which dominates the parsing complexity and can therefore not be a basis for a reasonable implementation. This parsing with the object grammar is called *indirect parsing*. On the other hand, there are approaches, called *direct parsing* which try to interleave the use of ID and LP rules. The approach taken in this paper is a result of augmenting this paradigm.

The concept of direct parsing was developed by Shieber (1984). He modifies Earley's algorithm (Earley 1970) to cope with ID/LP grammars. (Context–free) ID/LP grammars are defined by treating the formerly context–free rules as ID rules and by adding LP rules. The modification to the algorithm consists of two parts. The right hand side of a rule has no longer a fixed order, but is treated as a multiset. The predictor and the completer are limited to the LP acceptable structures by testing LP acceptability on the considered local trees, i.e. a nonterminal is extracted from the multiset and tested whether it may precede the remaining categories. The LP rules are taken and matched directly against proposed structures. Since this

*The research presented in this paper was sponsored by Teilprojekt A8 of the SFB 340 of the Deutsche Forschungsgemeinschaft. I want to thank Tom Cornell, Dale Gerdemann, Thilo Götz, Frank Richter and Klaas Sikkel for helpful discussion. Needless to say, all errors and infelicities that remain are my own.

[1] For unification grammars the number of rules may even be infinite.

proposal obviously does not suffice to handle HPSG style grammars, it has to be augmented to feature based grammars as is done in Seiffert (1991).

Unification based ID/LP grammars are defined by augmenting the domain of the nonterminals to feature structures and by formalizing when an LP rules applies. This happens in case the LP elements subsume two categories contained in a local tree. A violation occurs if the category which is subsumed by the first LP element follows the category which is subsumed by the second LP element. The Earley/Shieber parser is then used almost unchanged. The only change is that nonterminals are not longer identical, so that application of rules is determined by unification. But consider the grammar in figure 1 (taken from Seiffert (1991)).[2]

$$
\begin{array}{lcl}
\text{Lexicon} & = & \left\{ \begin{array}{lcl}
\left[\textsc{cat}\ d\right] & \rightarrow & h \\
\left[\textsc{cat}\ e\right] & \rightarrow & i \\
\left[\begin{array}{ll} \textsc{cat} & f \\ \textsc{f1} & \mathit{one} \end{array}\right] & \rightarrow & j \\
\left[\begin{array}{ll} \textsc{cat} & g \\ \textsc{f2} & \mathit{two} \end{array}\right] & \rightarrow & k
\end{array} \right\} \\
\text{ID–Rules} & = & \left\{ \begin{array}{lcl}
\left[\textsc{cat}\ a\right] & \rightarrow & \left[\begin{array}{ll} \textsc{cat} & b \\ \textsc{f} & \boxed{1} \end{array}\right], \left[\begin{array}{ll} \textsc{cat} & c \\ \textsc{f} & \boxed{1} \end{array}\right] \\
\left[\begin{array}{ll} \textsc{cat} & b \\ \textsc{f} & \left[\begin{array}{ll} \textsc{f1} & \boxed{2} \\ \textsc{f2} & \boxed{3} \end{array}\right] \end{array}\right] & \rightarrow & \left[\begin{array}{ll} \textsc{cat} & d \\ \textsc{f1} & \boxed{2} \end{array}\right], \left[\begin{array}{ll} \textsc{cat} & e \\ \textsc{f2} & \boxed{3} \end{array}\right] \\
\left[\begin{array}{ll} \textsc{cat} & c \\ \textsc{f} & \left[\begin{array}{ll} \textsc{f1} & \boxed{4} \\ \textsc{f2} & \boxed{5} \end{array}\right] \end{array}\right] & \rightarrow & \left[\begin{array}{ll} \textsc{cat} & f \\ \textsc{f1} & \boxed{4} \end{array}\right], \left[\begin{array}{ll} \textsc{cat} & g \\ \textsc{f2} & \boxed{5} \end{array}\right]
\end{array} \right\} \\
\text{LP–Rules} & = & \left\{ \begin{array}{lcl}
\left[\textsc{f1}\ \mathit{one}\right] & \prec & \left[\textsc{f2}\ \mathit{two}\right] \\
\left[\textsc{cat}\ b\right] & \prec & \left[\textsc{cat}\ c\right]
\end{array} \right\} \\
\text{Start Symbol} & = & \left[\textsc{cat}\ a\right]
\end{array}
$$

Fig. 1. Seiffert's example grammar

On input *ihjk* – which is not well formed – Seiffert's parser can not determine on the first pass whether the local tree ($\textsc{cat}\ b, (\textsc{cat}\ e, \textsc{cat}\ d)$) is well formed since the information that is relevant for the LP rule, namely that *one* has to precede *two*, is not available yet. Only after the local trees ($\textsc{cat}\ c, (\textsc{cat}\ f, \textsc{cat}\ g)$) and ($\textsc{cat}\ a, (\textsc{cat}\ b, \textsc{cat}\ c)$) have been constructed, the information becomes accessible through structure sharing. The nonlocal flow can be seen in the parse tree given in figure 2.[3] The information *one* and *two* comes from the lexical entries for f and g and is passed to e and d via c and b. Only then is the information that the value for F2 at e is *two* and that the value for F1 at d is *one* available and the local tree ($\textsc{cat}\ b, (\textsc{cat}\ e, \textsc{cat}\ d)$) has to be ruled out since the LP rule $[\textsc{f1}\ \mathit{one}] \prec [\textsc{f2}\ \mathit{two}]$ is violated.

This phenomenon makes it necessary for Seiffert to take the resulting parse tree and check it for any remaining LP violations. Clearly this is inefficient and not desired.

Although this paper is not concerned with linguistics, we nevertheless try to argue to some extent that the problem is indeed not only a technical one. Nonlocality is a well known phenomenon in natural language, but so far without any influence on word order. This section tries to sketch the fact that in a reasonably large fragment for German such nonlocal word order phenomena do exist. Since this is not the main goal of the paper, the presentation needs to be brief. It can be understood without an exact knowledge of the linguistic theories involved if one accepts the claims made during the discussion. But for a complete understanding of the details involved, we assume some knowledge of the analysis of partial verb phrases by Hinrichs and Nakazawa (1993), on word order by Lenerz (1977) and the empirical analyses by Richter and Sailer (1995).

Some data on word order in German sentences seem to suggest that there exists a connection between the main verb and the order of its complements. The example sentences in 1 do indicate

[2] To simplify notation, we are going to represent the trees of categories in the discussion of this example in term notation with just the category feature present.

[3] For readability reasons the arrows are drawn just for one value, namely *two*.

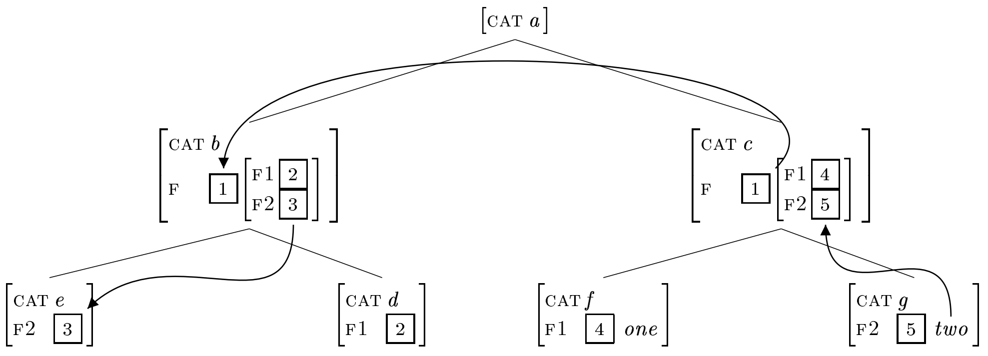

Fig. 2. The parse tree for input *ihjk and Seiffert's grammar (see figure 1)

the different behaviour of the verbs *geben* (to give) and *überlassen* (to leave).

1 Example Sentences

(1) a. Karl wird dem Kind das Geschenk geben wollen.
Karl will the child the present give want.
Karl will want to give the child the present.

b. *Karl wird das Geschenk $\begin{bmatrix}\textsc{rhem} & \textit{plus}\\ \textsc{case} & \textit{acc}\end{bmatrix}$ dem Kind $\begin{bmatrix}\textsc{rhem} & \textit{minus}\\ \textsc{case} & \textit{dat}\end{bmatrix}$ geben wollen.

(2) a. Karl wird das Geschenk dem Kind überlassen wollen.
Karl will the present the child leave want.
Karl will want to leave the child the present.

b. *Karl wird dem Kind $\begin{bmatrix}\textsc{rhem} & \textit{plus}\\ \textsc{case} & \textit{dat}\end{bmatrix}$ das Geschenk $\begin{bmatrix}\textsc{rhem} & \textit{minus}\\ \textsc{case} & \textit{acc}\end{bmatrix}$ überlassen wollen.

The *(a)* versions of the sentences do not exhibit any limitation on the order of the complements. The *(b)* versions reflect the fact that they do not allow the first NP to be rhematic in the sense of Lenerz (1977). With the verb *geben* this disallows the order of the accusative rhematic NP preceding the dative not rhematic NP, and in the case of *überlassen* the dative rhematic NP must not precede the accusative not rhematic NP. So the acceptance of the word order of the accusative and dative object seems to depend on the main verb and whether the first of those objects is to be rhematic. To express this in a grammar, one would have to ensure that there exists some connection from the main verb to the complements. We do not go into any detail how this connection could be formulated since it suffices to know that one has to exist. We do give a sketch for a parse tree for one of the example sentences in figure 3; V_a stands for an auxiliary verb, V_m for the main verb and V_k for a verbal complex. These parse trees rely on the analysis of German verb phrases in Hinrichs and Nakazawa (1993).

As one can see, the NP complements are not in one local tree with the main verb due to the necessary argument raising induced by the auxiliary in the verbal complex. But this is not sufficient to cause the problem presented previously. If one considers direct parsing of such a sentence, the local tree labeled VP in the figure 3 which has to decide on the LP acceptability of the order of the two NP complements is not completed unless the local tree of the verbal complex has been recognized. And by this, the necessary connection of the NP complements to the main verb can be validated so that the local tree containing the NP complements would not be completed.

A further complication is necessary to cause the problem – the topicalization of the verbal complex. The data to support the claim is not as sharp as for the non topicalized sentences,

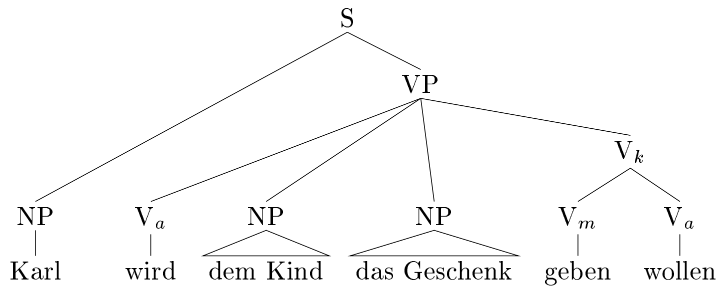

Fig. 3. A sketched parse tree for the sentence *Karl wird dem Kind das Geschenk geben wollen.*

although it is plausible that topicalization does not alter the behaviour of the main verbs concerning the word order of their complements. The examples are given below.

2 Example Sentences

(3) a. Geben wollen wird Karl dem Kind das Geschenk.

b. *Geben wollen wird Karl das Geschenk $\begin{bmatrix} \text{RHEM } \textit{plus} \\ \text{CASE } \textit{acc} \end{bmatrix}$ dem Kind $\begin{bmatrix} \text{RHEM } \textit{minus} \\ \text{CASE } \textit{dat} \end{bmatrix}$.

(4) a. Überlassen wollen wird Karl das Geschenk dem Kind.

b. *Überlassen wollen wird Karl dem Kind $\begin{bmatrix} \text{RHEM } \textit{plus} \\ \text{CASE } \textit{dat} \end{bmatrix}$ das Geschenk $\begin{bmatrix} \text{RHEM } \textit{minus} \\ \text{CASE } \textit{acc} \end{bmatrix}$.

As can be seen in the sketched parse tree in figure 4, the verbal complex is completely independent of the local tree containing the NP complements. This creates the desired constellation. If one considers right to left traversal of the input string for parsing, as is done for example in ALE (Carpenter 1993), the local tree whose mother is labeled VP has to be completed before the verbal complex is parsed. Therefore the information which main verb appears in the verbal complex is not yet known. The local tree would be LP acceptable although the whole parse tree may later contain information that the VP in question was not LP acceptable.

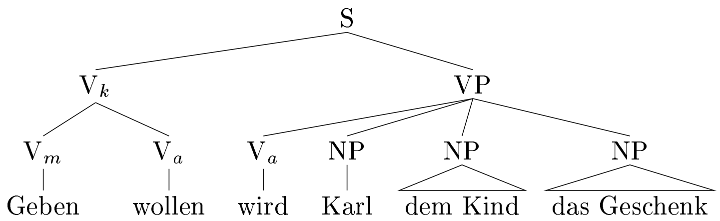

Fig. 4. A sketched parse tree for the sentence *Geben wollen wird Karl dem Kind das Geschenk.*

Although the presentation is somewhat sketchy, we showed that in natural language there may well be cases where the technical problem noted by Seiffert does indeed occur. Since all the mechanisms involved are present in current HPSG analyses of German and their interaction is complex, one may conclude that such phenomena are not easily detected by the grammar writer and can not necessarily be avoided. Therefore each parser dealing with unification based ID/LP grammars has to take care of those nonlocal phenomena.

2 Unification ID/LP Grammars

We are going to formalize unification ID/LP grammars very similarly to the approach taken in Sikkel (1993), because this seems close to a minimal definition which can be extended to almost all the versions of unification grammars. We use the notation and the feature machinery developed there, see chapter 8. This assumes disjoint, linearly ordered sets $Feat$ of features and $Const$ of constants. A standard representation of DAGs is used to define feature structures and from there it is straightforward to define a feature lattice. The definitions are augmented to deal with composite feature structures, i.e. multiply rooted feature structures and composite feature lattices to allow for the representation of phrase structure rules. The formalization assumes a context free backbone, but nothing in the treatment of the problem hinges on that.

3 Definition (Unification ID/LP Grammar) *A Unification ID/LP grammar is a tuple* $\mathcal{G} = (G, \Phi, \varphi_0, \varsigma_0, W, Lex)$ *such that:*

> $G = \langle N, \Sigma, ID, LP, S\rangle$ *is a (context free) ID/LP grammar.*
>
> $\Phi = \phi(Feat, Const)$ *is the lattice of feature structures.*
>
> $\varphi_0 : ID \to \Phi$ *is a function that assigns a composite feature structure to each rule in ID.*
>
> $\varsigma_0 : LP \to \Phi$ *is a function that assigns a composite feature structure to each rule in LP.*
>
> W *is the set of the word forms, i.e. lexical entries.*
>
> $Lex : W \to \wp(\Phi)$ *is a function that assigns a set of feature structures to each lexical entry.*

We write V *for* $N \cup \Sigma$, $\varphi_0(X_i)$ *for* $\varphi_0(X_0 \to X_1, \ldots, X_k)|_{X_i}$, $\varsigma_0(A)$ *for* $\varsigma_0(A \prec B)|_A$ *in case of the LP rules, and* $a, b, \ldots$ *for elements of* W. *It is assumed that* $\text{CAT} \in Feat$, $V \subseteq Const$ *and* $V \cap W = \emptyset$. *ID and LP are multisets of rules to allow different feature structures to occur with the same context free backbone. Take* $\varsigma(LP)$ *to be the set of all the* $\varsigma_0(X_i)$, *then* $\langle \varsigma(LP), \prec\rangle$ *is a transitive, asymmetric, irreflexive relation. Furthermore it is required that* $\forall\, 0 \leq i \leq k\, \varphi_0(X_i).\text{CAT} = X_i$ *for each production* $X_0 \to X_1, \ldots, X_k$ *and likewise for the LP rules and the lexical entries. Additionally we demand that* $\varphi(a).\text{CAT} \in \Sigma$. *The class of unification ID/LP grammars is denoted as* $\mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$.

As already mentioned in the previous section, an LP rule applies under subsumption. We assume a sequence of categories and check whether they are informative enough to determine applicability of the LP rules.

4 Definition (applies) *An LP rule* $A \prec B$ *applies to a sequence* $\sigma = \sigma_1 \ldots \sigma_n$, $\sigma_k \in V$ *iff there exist* i, j, $i \neq j$, $1 \leq i, j \leq n$ *such that* $\varsigma_0(A) \sqsubseteq \varphi(\sigma_i)$ *and* $\varsigma_0(B) \sqsubseteq \varphi(\sigma_j)$.

If no LP rule applies to a structure such that it causes a violation of the imposed order, it has to be LP acceptable.

5 Definition (LP acceptable) *A sequence* σ *is LP acceptable iff no LP rule applies to it with* $j < i$.

We overload the definition of LP acceptability in the sense that we use it on sets of sequences of feature structures as well. The need for this will become clear later.

A requirement for the definition of parse trees is the ability to generate ordered sequences from the multisets which represent the right hand sides of ID rules. Therefore we assume a permutation function on sequences of nonterminals called *permute*. We can define parse trees for ID/LP grammars by including the demand that each local tree conforms to the LP rules. We assume a standard tree formalization like the one from Partee, ter Meulen and Wall (1990) and augment it by a composite feature structure for the parse tree.[4]

6 Definition (parse tree) *A grammar* $\mathcal{G} = (G, \Phi, \varphi_0, \varsigma_0, W, Lex)$ *generates a parse tree* $\langle \tau, \varphi(\tau)\rangle$ *for the sentence* $a_1 \ldots a_n$ *iff*

> *the root is labeled with the initial symbol* S *of* G *and* $\varphi_0(S) \sqsubseteq \varphi(S)$;
>
> $\forall i\; 1 \leq i \leq n\; a_i \in W$ *and* $\varphi'(a_i) \in Lex(a_i)$ *such that* $\varphi'(a_i) \sqsubseteq \varphi(a_i)$;
>
> *for all subtrees* $\tau_i \in \tau$, *where* T_0 *immediately dominates* $T_1 \ldots T_n$, *there is a rule*

[4]Note that structure sharing is indicated with a plain $=$, whereas equality is denoted with $\doteq$.

$T_0 \rightarrow \alpha$ *in ID such that* $T_1 \ldots T_n \in permute(\alpha)$ *and*
$\forall i\ 1 \leq i \leq n\ \varphi_0(T_i) \sqsubseteq \varphi(T_i)$ *and*
the yield of the local tree τ_i *is LP acceptable.*

The parse trees defined thus are not necessarily adequately decorated, i.e. might have some features and values that do not derive from the application of the rules and lexical entries. The formalization of this can be taken from Sikkel (1993). We just demand that parse trees are minimal in this sense. This allows the specification of the language generated by a unification ID/LP grammar.

7 Definition (Language of the Grammar ($L(\mathcal{G})$)) *The language generated by* $\mathcal{G}$ *is defined as* $L(\mathcal{G}) = \{w = w_1 \ldots w_n |$ *there exists a parse tree* $\langle \tau, \varphi(\tau) \rangle$ *with yield* $w_1 \ldots w_n\ \&\ \varphi(\tau) \neq \bot\}$.

The parsing problem for the class of grammars in $\mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$ can now be stated as follows.

8 The Parsing Problem *Given a sentence* $a_1 \ldots a_n \in W^*$ *and a grammar* $\mathcal{G} \in \mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$, *find all the parse trees for the sentence with result* $\varphi(S)$.

This completes the definitions which present unification ID/LP grammars. But for the solution to the problem of nonlocal feature passing, we need some additional auxiliary definitions.

Since it may happen that a structure is not specific enough to determine LP acceptability, we have to formalize when these cases do occur. This is done by weak application. It tests for the success of a unification, i.e. whether the information could be there.

9 Definition (weakly applies) *An LP rule* $A \prec B$ *weakly applies to a sequence* $\sigma = \sigma_1 \ldots \sigma_n$, $\sigma_k \in V$ *iff there exist* $i, j,\ i \neq j,\ 1 \leq i, j \leq n$ *such that* $\varsigma_0(A) \sqcup \varphi(\sigma_i)$ *and* $\varsigma_0(B) \sqcup \varphi(\sigma_j)$.

Since we now can tell when there might be enough information to determine LP acceptability, it is straightforward to define possible LP violation. If a feature structure is subsumed by an element of an LP rule, it unifies with it as well. So we can not exclude the cases of application of an LP rule from our definition of weak application. So our definition has to presuppose that the structure is LP acceptable, under definition 5, i.e. that no LP rule applies to it and is violated. In that case we do not need to check for further (possible) violations since the sequence is ruled out anyway. If we could exclude *application* from *weak application*, this would not be necessary. And additionally, the weak application has to result in a violation if it is to be relevant at all.

10 Definition (possibly LP violated) *A sequence* σ *is possibly LP violated iff it is LP acceptable, but an LP rule weakly applies to it with* $j < i$.

Furthermore we have to add some bookkeeping device, the LP store. This is a list of the pending structures, i.e. those that are not specific enough to be determined yet[5]. The LP store is assumed to be part of each feature structure that is assigned to a rule. The formalism presented in Sikkel (1993) can be extended by demanding that every composite feature structure that represents a rule includes a list representation of the LP store. The following defines the union of two such stores. It will be used in the deduction steps.

11 Definition (Store union ($\bowtie$)) *The union of an LP store* Θ *with another LP store* Ω *with resulting LP store* Δ ($\varphi(\Theta) \bowtie \varphi(\Omega) \doteq \varphi(\Delta)$) *is recursivly defined as*
$\varphi(\Delta) \doteq \varphi(\Omega)$ *if* $\varphi(\Theta).\textsc{hd} \doteq e\_list$
$\varphi(\Delta) \doteq \varphi(\Theta).\textsc{tl} \bowtie \varphi(\Omega)$ *if* $\varphi(\Theta).\textsc{hd} \doteq \varphi(A) \wedge \varphi(A) \in \varphi(\Omega)$
$\varphi(\Delta).\textsc{hd} \doteq \varphi(A) \wedge \varphi(\Delta).\textsc{tl} \doteq \varphi(\Theta).\textsc{tl} \bowtie \varphi(\Omega)$ *if* $\varphi(\Theta).\textsc{hd} \doteq \varphi(A) \wedge \varphi(A) \notin \varphi(\Omega)$

3 An ID/LP Parsing Schema

Sikkel (1993) develops a framework called *parsing schemata* for comparing parsing algorithms by abstracting from control information and data structures contained in their specifications. We are going to use this framework because it is general enough to allow the specification of nontrivial algorithms without requiring implementation specific details. The presentation of a

[5]This is necessary since the feature logic does not support set valued feature structures.

parsing schema is done in three steps: firstly we define two kinds of parsing systems which then allow the specification of the parsing schema. A parsing system is restricted to a single given grammar and a single given sentence. For those we define a logical deduction system, called a parsing system.

12 Parsing System *A parsing system for some grammar G and input string $a_1, \ldots, a_n$ is a triple $\mathcal{P}(\mathcal{I}, \mathcal{H}, \mathcal{D})$ with $\mathcal{I}$ a set of items; $\mathcal{H}$ a finite set of items (not necessarily a subset of $\mathcal{I}$), the hypotheses; and $\mathcal{D}$ a set of deduction steps $\eta_1, \ldots, \eta_m \vdash \xi$ with $\eta_i \in \mathcal{I} \cup \mathcal{H}$ for $0 \leq i \leq m$ and $\xi \in \mathcal{I}$.*

There can be several kinds of items. But in the following, we are using an expanded version of the familiar items of an Earley or chart parser, e.g. $[A \rightarrow \alpha \bullet \beta, i, j]$. Now we extend this definition to an uninstantiated parsing system. This obviates the need for a fixed input string.

13 Uninstantiated Parsing System *An uninstantiated parsing system for a grammar G is a triple $\langle \mathcal{I}, \mathcal{H}, \mathcal{D} \rangle$ with $\mathcal{H}$ a function that assigns a set of hypotheses to each string $a_1, \ldots, a_n \in \Sigma^*$ such that $\langle \mathcal{I}, \mathcal{H}(a_1, \ldots, a_n), \mathcal{D} \rangle$ is a parsing system.*

It can easily be seen that in all our examples the function $\mathcal{H}$ is defined as follows: $\mathcal{H}(a_1, \ldots, a_n) = \{[a, i-1, i] \mid a = a_i, 1 \leq i \leq n \}$. Therefore we are not going to make a difference in the following between uninstantiated parsing systems and parsing systems.

14 Parsing Schema *A parsing schema for a class of grammars $\mathcal{G}$ is a function that assigns an (uninstantiated) parsing system to every grammar in $\mathcal{G}$.*

In the ID/LP parsing schema, we will interleave the two steps of generation via permute and testing via LP acceptability completely so that acceptance can be determined in just one traversal of the input. To achieve this, the deduction steps allowed by the direct parsing algorithm have to be modified. Augmenting Seiffert's and Shieber's definitions in the last section, we defined a relation that tells when an LP rule *weakly applies* to a sequence. Whenever this occurs, we add a sequence of feature structures to the LP store. This store may be thought of as information how further instantiation of the rule in question may be restricted. The restriction is not immediately obvious, but as soon as the rule is further instantiated, a test for LP acceptability is performed and the hypothesis discarded if the structure contained an LP violation. This added sequence of categories is structure shared with the sequence the parser assumes to be LP acceptable. So any changes that are made to the sequence via structure sharing are present on the LP store as well. In each deduction step the elements on the LP store have to be LP acceptable which achieves the desired effect. Since this LP violation can only happen in completion, this completion is ruled out, thereby preventing the parser from constructing invalid structures. The old structure is not discarded since it may be used in some other way which does not lead to an LP violation. Additionally, the algorithm has to percolate the LP stores via *store union* to ensure that all the edges do contain the pending LP information of their subcomponents.[6]

Before proceding to the presentation of the ID/LP parsing schema, a short remark on notation. The items have been augmented by the LP store Δ. We abbreviate in the following the notation of items by indexing them with lowercase greek letters and then refering to the index, i.e. $[A \rightarrow \alpha \bullet \beta, i, j, \Delta]_\xi$ is referred to as ξ. $\varphi(\xi)$ refers to the composite feature structure of the item, naturally excluding the string indices.

15 ID/LP Parsing System *A parsing system $\mathcal{P}(\mathcal{I}_{ID/LP}, \mathcal{H}_{ID/LP}, \mathcal{D}_{ID/LP})$ for a unification ID/LP grammar $\mathcal{G} = (G, \Phi, \varphi_0, \varsigma_0, W, Lex) \in \mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$ is defined by*

[6] These processes may be improved upon in an actual implementation, see Morawietz (1995) for the implementation in TROLL (Gerdemann, Götz and Morawietz forthcoming).

$$\mathcal{I}_{ID/LP} = \left\{ [A \to \alpha \bullet \beta, i, j, \Delta]_\xi \;\middle|\; \begin{array}{l} A \to \alpha\beta \in ID \\ 0 \le i \le j \\ \varphi_0(A \to \alpha\beta) \sqsubseteq \varphi(\xi) \\ \varphi(\xi) \not\doteq \bot \\ \alpha \text{ and } \Delta \text{ are LP acceptable} \end{array} \right\}$$

$$\mathcal{H}_{ID/LP} = \{ [a, j-1, j, \{\}] \mid \varphi(a) \in Lex(a_j) \}$$

$$\mathcal{D}^{Init} = \{ \vdash [S \to \bullet\gamma, 0, 0, \{\}]_\xi \mid \varphi(\xi) \doteq \varphi_0(S \to \gamma) \}$$

$$\mathcal{D}^{Scan} = \left\{ \begin{array}{c} [A \to \alpha \bullet \beta a \delta, i, j, \Delta]_\eta, [a, j, j+1, \{\}]_\zeta \vdash \\ [A \to \alpha a \bullet \beta\delta, i, j+1, \Delta]_\xi \end{array} \;\middle|\; \varphi(\xi) \doteq \varphi(\eta) \sqcup \varphi(\zeta) \right\}$$

$$\mathcal{D}^{Compl1} =$$
$$\left\{ \begin{array}{c} [A \to \alpha \bullet \beta B \delta, i, j, \Delta]_\eta, [B \to \gamma\bullet, j, k, \Omega]_\zeta \vdash \\ [A \to \alpha B \bullet \beta\delta, i, k, \Theta]_\xi \end{array} \;\middle|\; \begin{array}{l} \varphi(\xi) \doteq \varphi(\eta) \sqcup \varphi(\zeta) \\ \varphi(\Theta_\xi) \doteq \varphi(\Delta_\eta) \bowtie \varphi(\Omega_\zeta) \\ \varphi((\alpha B)_\xi) \text{ is not possibly LP violated} \end{array} \right\}$$

$$\mathcal{D}^{Compl2} =$$
$$\left\{ \begin{array}{c} [A \to \alpha \bullet \beta B \delta, i, j, \Delta]_\eta, [B \to \gamma\bullet, j, k, \Omega]_\zeta \vdash \\ [A \to \alpha B \bullet \beta\delta, i, k, \Theta]_\xi \end{array} \;\middle|\; \begin{array}{l} \varphi(\xi) \doteq \varphi(\eta) \sqcup \varphi(\zeta) \\ \varphi((\alpha B)_\xi) \text{ is possibly LP violated} \\ \varphi(\Theta_\xi) \doteq \varphi((\alpha B)_\xi) \bowtie \varphi(\Delta_\eta) \bowtie \varphi(\Omega_\zeta) \\ \varphi((\alpha B)_\Theta) = \varphi((\alpha B)_\xi) \end{array} \right\}$$

$$\mathcal{D}^{Pred} = \left\{ \begin{array}{c} [A \to \alpha \bullet \beta B \delta, i, j, \Delta]_\eta \vdash \\ [B \to \bullet\gamma, j, j, \{\}]_\xi \end{array} \;\middle|\; \varphi(\xi) \doteq \varphi(B_\eta) \angle \Psi_0(B) \sqcup \varphi_0(B \to \gamma) \right\}$$

$$\mathcal{D}_{ID/LP} = \mathcal{D}^{Init} \cup \mathcal{D}^{Scan} \cup \mathcal{D}^{Compl1} \cup \mathcal{D}^{Compl2} \cup \mathcal{D}^{Pred}$$

Note that our item definition demands that both the recognized sequence of categories and the LP store have to be LP acceptable. Therefore we do not have to include these demands in the deduction steps. The sequence of feature structures to the right of the bullet is treated as a multiset. This is indicated by picking a random element from it, i.e. both β and δ in the deduction steps can be empty or of any other cardinality. Furthermore it is important that in the definition of $\mathcal{D}^{Compl2}$ the new LP store is the result of the union of the old stores plus the sequence of categories found. This sequence is reentrant with the recognized one before the bullet. In the definition of $\mathcal{D}^{Pred}$, we included a restrictor ($\angle\Psi_0(B)$) to deal with the nontermination of the predictor. For the definition of a default restrictor, see again Sikkel (1993). Since this can be generated automatically, we did not have to include the definition of a restrictor in our grammar.

16 ID/LP Parsing Schema *A parsing schema* **P** *for the class of grammars* $\mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$ *is a function that assigns an ID/LP parsing system P to every grammar* $\mathcal{G} \in \mathcal{G}_{\mathcal{ID}/\mathcal{LP}}$.

Next, we are going to present the crucial steps an algorithm along the lines of the proposed schema takes to solve the problem of nonlocal feature passing. We simplify the notation in the sense that we do not differentiate between the context–free backbone and the corresponding feature structures any more. This gives us the following format for the edges. First comes the left hand side of the rule, followed by an arrow and the categories which have been recognized already, enclosed in square brackets. The dot separates those from the set of the categories yet to be found which is enclosed in curly brackets. Next are the start and end position of the recognized part of the edge. And last there is the LP store, again in curly brackets. The whole edge is enclosed in square brackets. We are going to write the values of tags, if there are any, only behind the crucial occurences.

The edges of interest are depicted in figure 5. We assume that the algorithm is supposed to recognize the input *ihjk* with the grammar given in figure 1. At some point in the computation

1. $\left[\ \begin{bmatrix} \textsc{cat}\ b \\ \textsc{f} & \begin{bmatrix} \textsc{f1} & \boxed{2} \\ \textsc{f2} & \boxed{3} \end{bmatrix} \end{bmatrix} \rightarrow \left[\boxed{\textsc{x}} \begin{bmatrix} \textsc{cat}\ e \\ \textsc{f2} & \boxed{3} \end{bmatrix} \begin{bmatrix} \textsc{cat}\ d \\ \textsc{f1} & \boxed{2} \end{bmatrix} \right] \bullet \{\}\ ,\ 0\ ,\ 2\ , \left\{ \boxed{\textsc{x}} \begin{bmatrix} \textsc{cat}\ e \\ \textsc{f2} & \boxed{3} \end{bmatrix} \begin{bmatrix} \textsc{cat}\ d \\ \textsc{f1} & \boxed{2} \end{bmatrix} \right\}\ \right]$

2. $\left[\ \begin{bmatrix} \textsc{cat}\ a \end{bmatrix} \rightarrow \left[\begin{bmatrix} \textsc{cat}\ b \\ \textsc{f} & \boxed{1} \end{bmatrix} \right] \bullet \left\{ \begin{bmatrix} \textsc{cat}\ c \\ \textsc{f} & \boxed{1} \begin{bmatrix} \textsc{f1} & \boxed{2} \\ \textsc{f2} & \boxed{3} \end{bmatrix} \end{bmatrix} \right\}\ ,\ 0\ ,\ 2\ , \left\{ \begin{bmatrix} \textsc{cat}\ e \\ \textsc{f2} & \boxed{3} \end{bmatrix} \begin{bmatrix} \textsc{cat}\ d \\ \textsc{f1} & \boxed{2} \end{bmatrix} \right\}\ \right]$

3. $\left[\ \begin{bmatrix} \textsc{cat}\ c \\ \textsc{f} & \begin{bmatrix} \textsc{f1} & \boxed{6}\ one \\ \textsc{f2} & \boxed{7}\ two \end{bmatrix} \end{bmatrix} \rightarrow \left[\begin{bmatrix} \textsc{cat}\ f \\ \textsc{f1} & \boxed{6}\ one \end{bmatrix} \begin{bmatrix} \textsc{cat}\ g \\ \textsc{f2} & \boxed{7}\ two \end{bmatrix} \right] \bullet \{\}\ ,\ 2\ ,\ 4\ , \{\}\ \right]$

*4. $\left[\ \begin{bmatrix} \textsc{cat}\ a \end{bmatrix} \rightarrow \left[\begin{bmatrix} \textsc{cat}\ b \\ \textsc{f} & \boxed{1} \end{bmatrix} \begin{bmatrix} \textsc{cat}\ c \\ \textsc{f} & \boxed{1} \begin{bmatrix} \textsc{f1} & \boxed{2}\ one \\ \textsc{f2} & \boxed{3}\ two \end{bmatrix} \end{bmatrix} \right] \bullet \{\}\ ,\ 0\ ,\ 4\ , \left\{ \begin{bmatrix} \textsc{cat}\ e \\ \textsc{f2} & \boxed{3}\ two \end{bmatrix} \begin{bmatrix} \textsc{cat}\ d \\ \textsc{f1} & \boxed{2}\ one \end{bmatrix} \right\}\ \right]$

Fig. 5. Edges from the chart for input word *ihjk* and Seiffert's grammar (see figure 1)

it has to create the local tree represented by the edge in 1. Note that since the algorithm cannot decide on the LP acceptability of the recognized structure and because it is possibly LP violated, it adds the sequence of categories to the LP store via structure sharing (tag $\boxed{\textsc{x}}$).[7] This information is percolated to further edges which rely on the computation of 1, for example the one in 2. Independently we construct the local tree represented by the edge in 3. At some point it becomes necessary for the algorithm to complete those two edges 2 and 3. But this can not be done as shown by the edge in 4. It is included in this chart only for the reason to illustrate the point. It would *not* be constructed by the algorithm. The reason for this is as follows. The completer would try to build the edge from the edges 2 and 3 by moving the recognized feature structure with the cat feature c to the list to the left of the dot. This feature structure is the result of the unification between those two feature structures from the edges 2 and 3 which have the category c. The unification forces identity between the reentrancies $\boxed{2}$ and $\boxed{6}$ and $\boxed{3}$ and $\boxed{7}$ respectivly. This results in the values *one* and *two* on the elements of the LP store which violates the LP rule $[\textsc{f1}\ one] \prec [\textsc{f2}\ two]$. Since it is not possible to construct a parse tree in another way, the sentence is not in the language of Seiffert's grammar.

4 Conclusions

In this paper, we defined unification ID/LP grammars and gave instructions for a parsing algorithm, thereby solving the problem of nonlocal feature passing. This enables linguists to write ID/LP grammars and gives the computational linguists a clear formalization and the means to directly parse such grammars in one pass.

Our approach presents a local solution to a nonlocal phenomenon. It uses the existing mechanism of the test for LP acceptability to detect the violations as early as they can be discovered. No complex backtracking into the chart or of the whole process is necessary, since it is not the local tree where the LP violation occurs that is seen as being invalid, but rather the combination of the information from this local tree with another local tree. It is acceptable to combine the local tree with another structure in a way that does not lead to failure.

Furthermore, the solution is neutral toward processing direction, i.e. bottom up versus top down, and processing mode, i.e. parsing versus generation. The way the solution is presented here is for a top down parser, but nothing in the way the problem is handled requires any of the special mechanisms involved with top down parsing. In fact, the first prototype to be implemented was a bottom up parser. One can just leave out the deduction steps that represent prediction. And since Earley's algorithm may be used for generation as well as for parsing (Gerdemann 1991), it seems straightforward to incorporate our treatment of the ID/LP

[7] This tag is special in the sense that it denotes sharing of a sequence rather than a single feature structure. So the tag is a simplification for a number of tags which would indicate the sharing of all the roots of the categories involved.

formalism for generation.

Nevertheless, some open questions remain. Some are induced by linguistic needs and some by formal considerations. Although we are not really concerned with it, from a linguistic point of view there are some demands that have not been met by the proposed approach. In some linguistic analyses, only binary branching structures are considered, for example in Uszkoreit (1984). This would lead to problems with direct parsing from a linguistic point of view because in our approach only local trees can be ordered by LP rules. If one is dealing with these binary structures, the domain of application of LP rules has to be enlarged, for example to something along the lines of a head domain, as in Engelkamp, Erbach and Uszkoreit (1992). In our approach, this would have to be done by some extra mechanism that collects the categories in the list under the recognized feature until the projection becomes itself the argument.

A technical problem that remains to be tackled is the incorporation of set–valued feature structures into the formalization. This would simplify the definitions since we would not have to use store union in the combination of two edges but could simply unify them. But since this is a new issue and an open problem, we cannot solve it here.

The implementation of the schema for the typed feature system TROLL (Gerdemann et al. forthcoming) shows that a system may use this kind of approach to provide the means for a concise representation of ID/LP grammars. The other approaches found in the literature such as for example Engelkamp et al. (1992) who incorporate a treatment of ID/LP into an HPSG style formalism by altering the ID schemata and by the creation of principles, cannot deal with the problem of nonlocal feature passing. Since in their approach the lexical items determine what must or must not appear to their left and right respectivly, they suffer from further instatiation of LP relevant information in the same way that the parser proposed by Seiffert does. All other proposed treatments of the ID/LP paradigm try to incorporate it into the formalism of HPSG directly. This reflects the different viewpoints between a rigid (denotational) formalization of HPSG and (operational) considerations which use unification grammars. Our approach defines ID/LP grammars in the latter sense.

References

Carpenter, B. (1993). ALE: The attribute logic engine, *User manual*, Laboratory for Computational Linguistics, Philosophy Department, Carnegie Mellon University, Pittsburgh, PA 15213.

Earley, J. (1970). An efficient context-free parsing algorithm, *Comm. ACM 13.2* pp. 94–102.

Engelkamp, J., Erbach, G. and Uszkoreit, H. (1992). Handling linear precedence constraints by unification, *ACL Proceedings, 23th Annual Meeting*, pp. 201—208.

Gazdar, G., Klein, E., Pullum, G. K. and Sag, I. A. (1985). *Generalized Phrase Structure Grammar*, Harvard University Press, Cambridge, Massachusetts.

Gerdemann, D. D. (1991). *Parsing and Generation of Unification Grammars*, PhD thesis, University of Illinois.

Gerdemann, D. D., Götz, T. W. and Morawietz, F. (forthcoming). *Troll – Type Resolution System – Fundamental Principles & User's Guide*, Universität Tübingen.

Hinrichs, E. W., Meurers D. and Nakazawa, T. (1993). Partial-VP and Split-NP Topicalization in German - An HPSG Analysis -, *Arbeitspapiere des SFB 340 58*, Universität Tübingen.

Lenerz, J. (1977). *Zur Abfolge nominaler Satzglieder im Deutschen*, TBL Verlag Gunter Narr, Tübingen.

Morawietz, F. (1995). Formalization and Parsing of Unification–Based ID/LP Grammars, *Arbeitspapiere des SFB 340 68*, Universität Tübingen. (Available in the World Wide Web under "http://www.sfs.nphil.uni-tuebingen.de/~frank/").

Partee, B. H., ter Meulen, A. and Wall, R. E. (1990). *Mathematical Methods in Linguistics*, Vol. 30 of *Studies in Linguistics and Philosophy*, Kluwer Academic Publishers, Dordrecht/Boston/London.

Pollard, C. J. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*, University of Chicago Press and CSLI Publications.

Richter, F. and Sailer, M. (1995). *Remarks on Linearization*, Master's thesis, Universität Tübingen.

Seiffert, R. (1991). Unification–ID/LP grammars: Formalization and Parsing, *in* O. Herzog and C.-R. Rollinger (eds), *Text Understanding in LILOG*, Springer, Berlin, pp. 63–73.

Shieber, S. M. (1984). Direct Parsing of ID/LP Grammars, *Linguistics and Philosophy* **7**: 135–154.

Sikkel, K. (1993). *Parsing Schemata*, PhD thesis, University of Twente, Enschede, The Netherlands.

Uszkoreit, H. (1984). *Word Order and Constituent Structure in German*, PhD thesis, University of Texas, Austin. Published in revised form in 1987 as Lecture Notes No. 8, CSLI Lecture Notes Series.