*THE PROPER TREATMENT OF WORD ORDER IN HPSG**

KAREL OLIVA

Department of Computational Linguistics
University of Saarland
Im Stadtwald
W-6600 Saarbrücken 11
Federal Republic of Germany
e-mail: karel@coli.uni-sb.de

Abstract:

This paper describes a possibility of expressing ordering constraints among non-sister constituents in binary branching syntactic structures on a local basis, supported by viewing the binary branching structure as a list (rather than a tree) of constituents within HPSG-style grammars. The core idea of such a description of ordering is constituted by creating a type lattice for lists. The possibilities of expressing different approaches to word order in the framework are briefly discussed, exemplified and compared to other methods.

In the standard immediate-constituent based approaches, the "free" word order[1] is described either directly in the phrase-structure (PS) rules, which thus express simultaneously both dominance (mother/daughter) relations and precedence (ordering) relations between syntactic categories, or, in more recent formalisms such as GPSG or HPSG, by the linear precedence (LP) rules creating a separate component of the grammar, whose other component is the set of immediate-dominance (ID) rules. In both these cases, the ordering constraints are limited to sister constituents, i.e. they are strictly local.

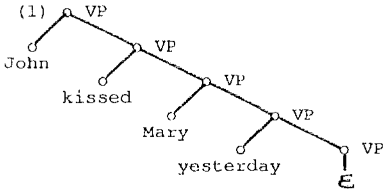
One of the problems of both these variations of the standard PS-approach is the description of adjuncts (free modifiers). On the usual assumption that their number per clause is in principle not limited (though finite for a particular clause), an approach to their ordering presupposing them to be all sister constituents must necessarily presuppose also an (at least potentially) infinite set of generative rules (e.g., a set induced by a Kleene star used in a "basic" variant of one of the rules).

In languages whose grammar allows for more word order freedom than English[2], it is often the case that adjuncts and complements of a head (typically, but not solely, of a finite verb) can be freely intermixed, which makes the approach where the locality of LP constraints forces the head as well as its modifiers (both complements and adjuncts) be expanded as sisters still less attractive.

Another possibility of description of word order is the "topological" approach used predominantly in more traditionally oriented German linguistics. Applied to German, this approach divides a clause into several word order "fields" ("Vorfeld", "Mittelfeld", "Nachfeld", "linke/rechte Satzklammer") whose mutual position is fixed, and studies mainly the word order regularities within these "fields". Though a lot of work has been done and many valuable insights presented within this paradigm, seen from the viewpoint of computational linguistics this approach has the fatal disadvantage that it is extremely difficult to formalize within the standard frameworks (e.g., none of the "fields" with the possible exception of "Vorfeld" creates a constituent in any usual sense etc.).

As an alternative to the two basic approaches mentioned above (as a modification of the first one, in fact), the description based on binary right-branching structures has been proposed independently in several works concerned with languages exhibiting a considerable share of free word order: in (Uszkoreit,1986) for the description of verb-final clauses in German, in (Gunji, 1987) for Japanese and in (Avgustinova and Oliva,1990) for (mainly) Slavic languages. However, the price paid for the removal of some problems, mainly, the free intermixing of heads, complements and adjuncts, of the abovementioned more standard descriptions is rather high - at least two problematic points arise due to the strict binarity of the structure. The first of them is the fact that in binary structures no LP-rules relying on the relation "being sister constituent" can be used for ordering heads, complements and adjuncts in cases this is required, since these are not sisters any more. The second problematic point can be seen at best at the variant of the formalism given in (Avgustinova and Oliva,1990) - the occurrence of the phonologically empty rightmost element of the branching[3] (cf. the structure (1) for the string "John kissed Mary yesterday").

(1)



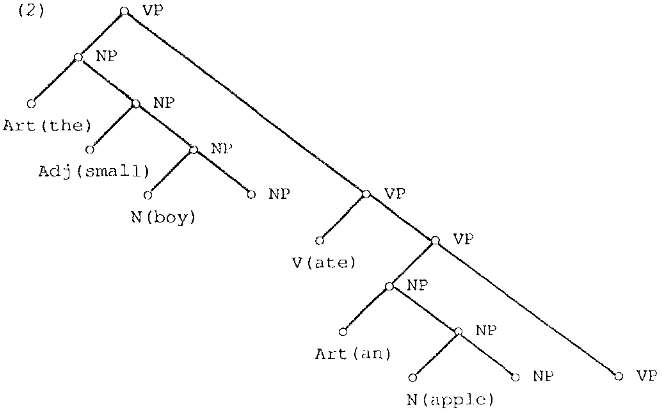
The former problem concerning word order is in the majority of the binary-branching approaches (as far as they are at all concerned with it) solved by introducing word order mechanisms which are either of non-local nature or which burden the syntactic categories (understood as feature bundles) with otherwise unmotivated features used solely for the purpose of imposing ordering constraints (and most often, with a combination of the two). Neither approach is more fortunate than the other - non-locality is surely an unwished phenomenon in the description, and the presence of special ordering features in the categories is hardly better, i.a. also because order is a property of the syntactic structure (made of categories) rather than of the categories themselves[4].

This paper will try to show that in spite of the abovementioned reservations the "binary branching" can be a correct and fruitful approach to syntactic description if seen from a slightly different viewpoint. In order to get the proper perspective, let us observe the "binary branching" structure for the example sentence "The small boy ate an apple" shown in (2).

There are several things to be taken into consideration here.

The most obvious among them is the division of the structure into "levels" - contiguous sequences of nodes with identical marking. Thus one

(2)

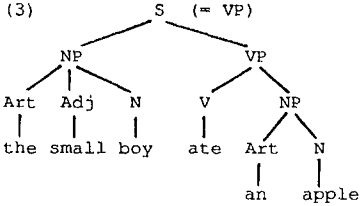


"VP-level" and two "NP-levels" are to be clearly seen, each having a distinguished element at its end (the phonologically empty element).

Further, it can be observed that each of the "levels" has one (and only one) other distinguished element somewhere in a non-final position - it is the ***V*** element for the "VP-level" and the ***N*** elements for the "NP-levels", in other words each level has a head.

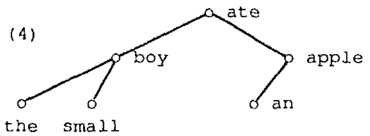
It is also worth remarking that the levels of the binary branching have a direct relation to more usual approaches. Thus, the standard immediate-constituent tree (3) for the sentence from (2) can be obtained by factorizing (collapsing) the NP nodes from the respective NP-levels into a single one, and by factorizing all the VP-nodes except for the uppermost "sentential" one.

(3)



The dependency tree (4) of the same sentence can be then obtained by collapsing all nodes of a level plus its head into a single node.

(4)



The most striking observation concerning the syntactic structures of the kind exemplified in (2), however, is the nature of their data type: showing a strict binary right-directed branching and having a distinguished (by its phonological emptiness) node as their final element, they are in fact nothing but *lists*[5].

Adopting this view brings along several advantages:

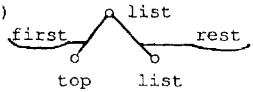
- first, the syntactic structure is strictly uniform - and also simpler than the general tree structure, with all (mainly practical) consequences following from this

- second, the overall usage of lists (whose members may be lists again) brings back the notion of locality of syntactic description - each list used in the structure (i.e., each "level" of the structure as discussed above) constitutes a local domain, creating thus also a natural area of application of local constraints (such as subcategorization, linear precedence etc.)

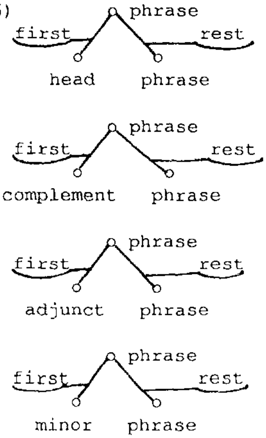
- third, such an approach allows for merging both the syntactic and the topological approaches in a single formal description, keeping, however, the two components clearly separated - the categorial information being expressed by means of (syntactic and other) features and their bundles (attribute-value matrices), the topological information being expressed by means of refinement of kinds of lists and their elements and sublists.

Thus, given fairly usual assumptions about the nature and function of constituents in a phrase, the general type[6] for nonempty lists from (5) is to be split into subtypes shown in (6) (where ***minor*** covers constituents made of complementizers, particles etc.).

(5)



(6)



In practice, even more delicate division is needed according to kinds of phrases used and according to the nature of modifiers these phrases allow. Introduction of more fine-grained subtypes may be needed also for the final element of a list (usually ***nil***); the respective subtypes should mirror the kinds of phrases used as functions of the "levels" of the syntactic structure, giving thus rise, e.g., to types ***end_of_np***, ***end_of_vp*** etc.

Using a different form of structural representation enforces also using different form (but not different background intuitions) of rules and principles of the grammar, all of them corresponding to the types of lists as introduced in the immediately preceding text.

Thus, the Head Feature Principle (HFP) is to be expressed as a conjunction of two implications[7] (rather than a single implication, combining conjunctively with other principles of the grammar), one describing the case where the first element of the respective nonempty[8] list is the head of the respective phrase ("level"), the other one describing the rest of the cases.

(7) Head Feature Principle

```
 [first: [head ]]
  => [synsem:cat:head: |1|           ]
     [first:synsem:cat:head: |1|     ]
     [rest:synsem:cat:head: |1|      ]
&
 [first: [not(head) ]]
  => [synsem:cat:head: |1|           ]
     [rest:synsem:cat:head: |1|      ]
```

Assuming the version of HPSG using sets (rather than lists) as values of the feature ***subcat***, the Subcategorization Principle has to consist of four implications, each for a particular configuration in the syntactic list[9]. The first part describes the impact of an expansion of a complement, the second the impact of an expansion of a head (consisting just in copying the subcategorization of the head into a special head feature ***head_subcat***, with the aim of inheriting the information about the subcategorization of the head constituent into the final element of the respective list via the HFP), the third the impact of an expansion of an adjunct or of a minor category, and the last one expresses the requirement that the subcategorization of the final element of the phrase, covering no syntactic material, be equal to the subcategorization of the head of the phrase. (The effect of the second and the fourth implication taken together is worth comparing with the above-mentioned condition from works by Uszkoreit and Gunji, namely that the verb - the source of the subcategorization - stand at the end of the clause.)

(8) Subcategorization Principle

```
 [first: [complement ]]
  => [synsem:subcat: |1|                    ]
     [first: |2|                            ]
     [rest:synsem:subcat: |1| ∪ {|2|}       ]
&
 [first: [head ]]
  => [synsem:cat:head:head_subcat: |1|]
     [first:synsem:subcat: |1|        ]
&
 [first: [not(head) & not(complement) ]]
  => [synsem:subcat: |1|     ]
     [rest:synsem:subcat: |1|]
&
 [nil ]
  => [synsem:[subcat: |1|                    ]]
     [       [cat:head:head_subcat: |1|      ]]
```

Assuming further a phonological principle stating that the phonology of constituents of the type ***nil*** (and of all its subtypes) is empty while phonology of all other constituents is the combination[10] of phonologies of their ***first*** and ***rest*** subconstituents, this approach allows for reduction of the number of grammar schemata ("rules") describing the categorial structure to one (similarly as in Gunji,1987) having the gross shape shown in (9).

```
(9)    [first: []]
       [rest:  []]
```

The word order constraints, on the other hand, can be expressed within the hierarchy of sorts of lists used in the system, by means of which the ordering information is not only kept separated from the categorial one, but is also formulated in local domains (each constituted by a list) only.

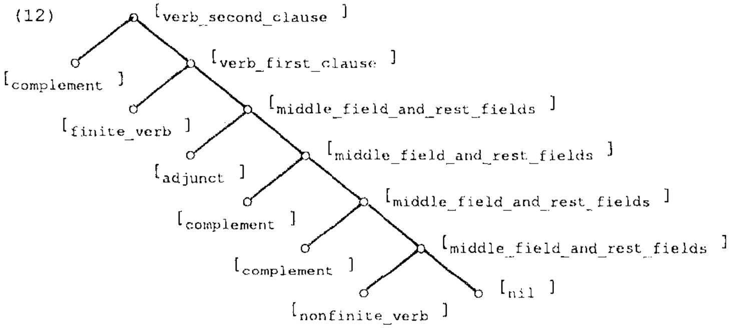
The practical usage of the idea of using the sort hierarchy of lists for the purpose of expressing word order constraints will be now illustrated on an example. In this example, the symbol "==" will be used for defining the *type hierarchy*. The type standing on the left-hand side of the "==" will be a *supertype* of the type standing on its right-hand side. As the example proper, let us take a slightly simplified system of German word order as used in the "field"-based approach, and let us assume for the moment that the sorts ***finite_verb***, ***nonfinite_verb***, ***complement*** and ***adjunct*** are primitives (though, obviously, in reality they are not). The description of the word order of the clause then may look like as shown in (10).

```
(10) [clause ] == [verb_first_clause ]
                v [verb_second_clause ]
                v [verb_last_clause ] .

     [verb_first_clause ] == [first: [finite_verb ]                      ]
                             [rest : [middle_field_and_rest_fields ]     ] .

     [verb_second_clause ] == [first: [forefield ]           ]
                              [rest : [verb_first_clause ]   ] .

     [verb_last_clause ] == [first: [verbal_modifier ]  ]
                            [rest : [verb_last_clause ] ]
                          v
                            [first: [finite_verb ] ]
                            [rest : [nil ]         ]

     [forefield ] == [verbal_modifier ] .

     [middle_field_and_rest_fields ] ==
                            [first: [verbal_modifier ]                  ]
                            [rest : [middle_field_and_rest_fields ]     ]
                          v
                            [nil ]
                          v
                            [first: [nonfinite_verb ] ]
                            [rest : [after_field ]    ] .

     [after_field ] == [nil ]
                     v
                       [first: [verbal_modifier ] ]
                       [rest : [after_field ]     ] .

     [verbal_modifier ] == [complement ]
                         v [adjunct ] .
```

The first definition in (10) describes the fact that a clause is either a verb-first clause, a verb-second clause or a verb-last clause. The next three definitions describe the word order within these kinds of clauses. The definitions of verb-first and verb-second clauses are quite simple, specifying only the types of the ***first*** and ***rest*** features of the respective syntactic lists. The definition of verb-last clauses expresses the fact that they can consist either of a verbal modifier followed by (the rest of the) verb-last clause, or of a finite verb, which cannot be followed by any syntactic material[11]. The last four definitions express actually the "field" approach to the German sentence. The first of them states that the forefield consists of a verbal modifier, which, in turn, is defined as being either a complement or an adjunct (in the last definition). The specification of middle field (and contingent following parts of the clause) says that the it can contain first of all any verbal modifier followed by (the rest of the) middle field or that it can be empty or that it can contain a nonfinite verb followed by an afterfield. Finally, the afterfield is defined either as empty or as containing a verbal modifier followed by (the rest of the) afterfield. Some clarification of the general idea should be brought in by the structure (12) for the sentence (11). Here, as well as in the structures that follow, only the most specific deducible sorts are given for a type (e.g., with the constituent ***Hans*** the sort ***complement*** is used rather than the sort ***forefield,*** because ***complement*** is more specific than ***forefield***).

(11) Hans hat gestern Maria ein Buch gegeben.

(12)



The previous example showed a relatively simple case where the number of elements ("fields" of the clause) to be ordered was low and more or less given in advance, and their ordering absolute (e.g., forefield first, finite verb second, middlefield third etc.). However, the descriptive power of the approach is not limited to this: cases where the number of elements is not given beforehand and their ordering is not absolute can be coped with, too, as well as cases of word order combining the two kinds of requirements. For more details see (Oliva, 1992).

The principal achievement of the approach presented is the (re)introduction of locality into binary branching structures allowing for replacing the more standard but in this case unsuitable concept of ordering constraints holding for sister constituents by a very similar concept of ordering constraints holding for a list of constituents. However, the introduction of lists allows also for easiness of expressing different other techniques of describing word order and its variations, such as the "topological" approach discussed above[12] or the "systemic ordering" as worked out by Prague linguists (e.g., Sgall et al.,1986) etc. Worth consideration is also the relation of this approach based on typing of lists of surface constituents (expressing thus i.a. also their obliqueness hierarchy) to the "<<"-type of LP-rules of the standard HPSG, which, unlike the approach discussed, force obliqueness of complements to be expressed repeatedly within the subcategorization list of each head.

The applicability of the method of description of word order as discussed in this paper has been proved by using it successfully in an experimental grammar of German developed in the STUF '91 formalism within the *LiLog* project.

Acknowledgement

I am thankful to Tania Avgustinova, Judith Engelkamp, Gregor Erbach, Eva Hajičová and Petr Sgall for most valuable comments on the first draft of this paper.

Bibliography:

Avgustinova T. and K. Oliva: Syntactic Description of Free Word Order Languages, in: Proceedings of Coling '90, Helsinki 1990

Gazdar G.,E.Klein,G.Pullum and I.Sag: Generalized Phrase Structure Grammar, Basil Blackwell, Oxford 1985

Gunji T.: Japanese Phrase Structure Grammar, Reidel, Dordrecht 1987

Oliva K.: Word Order Constraints in Binary Branching Syntactic Structures, to appear in the report series of CLAUS (Computational Linguistics at University Saarbrücken) 1992

Pollard C. and I.Sag: Information Based Syntax and Semantics vol.1: Fundamentals, CSLI Lecture Notes No. 13, CSLI, Stanford, California 1987

Sgall P., E.Hajičová and J.Panevová: The Meaning of the Sentence in Its Semantic and Pragmatic Aspects, Academia, Prague 1986

Uszkoreit H.: Linear Precedence in Discontinuous Constituents: Complex Fronting in German, CSLI Report No.47, CSLI, Stanford, California 1986

Footnotes:

* Needless to say, hopefully, that this title should not be taken all that seriously ...

1 In fact, free *constituent* order - but I shall stick to the traditional terminology in this paper.

2 Though even the English word order is not that rigorous as is often assumed - especially as to the position of different adjuncts.

3 This is not to say that this problem is not latent also in the approaches presented by Uszkoreit and Gunji - they just use the clause-final position of the verb in German and Japanese (i.e., a phenomenon from the fixed-word-order sphere) which helps to cover it.

4 Though in HPSG the formal difference between the two is removed due to the existence of the feature ***d_trs*** (daughters), the intuitive difference of course remains.

5 For this reason, the term **syntactic list** will be used later in this paper as an equivalent of the term **binary branching syntactic structure**.

6 In the present paper, the term **type** is used for what is usually called also a *feature structure*, an *attribute-value matrix* etc. The term **sort** will be used for *kinds* of feature structures, i.e. for what is sometimes used to be called *type* (of a feature structure). This convention will be used consequently and should not, thus, cause misunderstanding. Both types and sorts create their respective lattices, in principle independent from each other. This allows for using operators ***&***, **v** and ***not*** for creating unification, disjunction and complement, respectively, of types and sorts. In the following examples, types will be given as attribute-value matrices enclosed in square brackets, and sorts as subscripts in italics of these brackets.

7 Remember that the principle in the form of an implication applies only if the left-hand side of the implication unifies with the structure the principle should apply on; note also that in the particular case formulated here, the HFP could have been simplified into a conjunction of a non-implication and an implication parts.

8 No form of HFP applies on the empty list - no inheritance of features between mother and head daughter can occur there, obviously.

9 Notwithstanding the particularization (four implications instead of one in standard HPSG), this still should be treated as a principle - it is a generalization holding across kinds of phrases (NP's, VP's etc.).

10 Typically, but not necessarily, concatenation.

11 Two short remarks seem to be needed on this spot. First, here the fact that in German the clause cannot be constituted by a finite verb alone, is to be coped in other parts of the grammar (e.g., by the subcategorization of the verb). Second, the fact that no afterfield is allowed in verb-last clauses in this example is an arbitrary decision, having little to do with the general expressive power of the presented approach, and even less with grammar of "real" German.

12 Of particular importance on this spot seem to be the facts that on the approach sketched it is naturally possible to speak about, e.g., the *middlefield*, giving to this term also a clear-cut formalized treatment but without forcing it to occur as a true *constituent* in the description, as well as the possibility to specify the position of the verb in verb-first and verb-second German clauses without retorting to any kind of "movement" mechanism (e.g., to ***SLASH***).