Topological frames in sign-based grammars

Lars Ahrenberg
Linköping

Abstract

The paper presents some ideas on how topological frames can be integrated in HPSG-like grammatical descriptions and be used for parsing. Phrase structure is taken to be purely hierarchical and is represented by the special feature DTRS. The topological frames account for basic word order constraints of major categories, while linear precedence rules account for word order constraints within the positions of a topological frame.

Introduction

In a context-free phrase structure grammar, whether augmented with features or not, a rule expresses simultaneous constraints on hierarchical and sequential relationships. Gazdar et al. (1985) showed how general rules of word order (LP-rules) could be formulated independently of hierarchical relations and, together with a set of unordered phrase structure rules (ID-rules), define a phrase structure grammar of a special form. The local tree in (1) is licenced either by the rewriting rule (2) or by the ID- and LP-rules of (3a,b).

(1) $_{VP}$[V NP PP]

(2) VP → V NP PP

(3) (a) VP → V, PP, NP; (b) V < NP, V < PP, NP < PP

Pollard & Sag (1987) developed these ideas by showing how general rules of (unordered) phrase structure can be stated within a formalism employing typed feature structures. Sequential relationships are still handled by LP-rules, but have a different domain; they no longer order dominated constituents directly, but apply to values of the special attribute PHON. The phonological expression associated with a mother must then be some permutation of the phonological expressions associated with the daughters that respect all LP-rules. An HPSG-like grammatical representation of (1) is shown in (4), where the value of PHON is determined by analogs of the LP-rules in (3b).

Proceedings of NODALIDA 1993, pages 9-17

```
(4)    vp;
       PHON = <1 2 3>
       SYN:LOC:SUBCAT = <x>
       DTRS:HEAD = [verb; PHON = 1, SYN:LOC:SUBCAT =
       <x[np], y[np], z[pp]>]
       DTRS:CDTRS: = <y[PHON = 2], z[PHON = 3]>
```

There are problems, however, for grammars relying on LP-rules as the sole means for stating word order constraints. Languages with discontinuous constituents, such as the Scandinavian languages, and especially German, pose difficulties. There have accordingly been many proposals to augment LP-rules in various ways. Reape (1989) proposes a more complex combinatoric operation, sequence union, which allows access to non-immediate daughters of a constituent, while Engelkamp et al. (1992) propose to widen the domain of LP-rules to what they call head-domains, i.e. sets of constituents consisting of a lexical head with all its complements and adjuncts. In this paper I propose instead to restrict the use of LP-rules to smaller domains, called clusters, while augmenting the grammar with another device to handle word order regularities: the topological frame. The frames encode word order regularities that are valid for a class of constituents. They can basically be thought of as formalizations of the topological schemas used by Diderichsen (1962) and several other linguists working in his tradition. A cluster can similarly be seen as a sequence of constituents occuring within a specific position (or field) of a frame.

For reasons of space the full motivations and implications of this proposal cannot be dealt with here, though see Ahrenberg (1990) for some of the motivations. Instead I will develop a small, illustrative grammar fragment to make the proposal more tangible.

Elements of the grammar

The language fragment used is small and simplified in many respects. What I propose is quite compatible with the general assumptions of HPSG, however, apart from the account of word order regularities; I assume that it is necessary to restrict the domain of word order rules in languages like Swedish and German to types. This is after all quite a natural assumption to make in a theory assuming grammars to be organized as type hierarchies. In particular, topological frames apply to phrase types while LP-rules apply to clusters.

The basic elements of the grammar are signs and clusters. While both elements have overt expressions, indicated by the attribute STRING, only signs carry substantial linguistic information, indicated by the attribute FEATS. A cluster is basically a sequence of signs, indicated by the attribute ITEMS, which is connected and contracts specific sequential relations w r t other signs and clusters. It is often, though not always, the case that items of a cluster have a common grammatical status. Some putative examples of clusters are:

- The complements of a head, e.g *put / the books on the table*;
- A sequence of adjacent modifiers, e.g. *a / big black / building*

Signs are either phrasal or lexical (i.e. words). A phrase is distinguished from a word by having a constituent structure indicated by the attribute DTRS. The value of DTRS is a feature structure where attributes such as HEAD, SUBJ (for subject), CDTRS (complements other than subjects) and ADTRS (adverbials and adjuncts) appear. A phrase also has a structure imposed on its expression, which is registered under the attribute PATTERN. The value of PATTERN is a topological frame, i.e. a finite list of elements constructed out of strings and dominated patterns. The value of the attribute STRING is a list of strings with no embedded lists (cf. PHON of Pollard & Sag, 1987). The value of FEATS is a feature structure where we find attributes representing morphosyntactic properties such as MOOD and SUBCAT (subcategorization). A partial description of the sentence *Johan lade väskan på bordet* (John put the bag on the table) can be found in (5).

It should be observed that the phrase structure shows more branching than the topological structure. Although a verb phrase (a predicate) is part of the phrase structure, there is no distinct topological frame for it. Instead, its topology is identified with that of the clause as the two paths, PATTERN and DTRS:HEAD:PATTERN, share the same frame.

(5) main-clause;[1]
STRING = <1:Johan 2:lade 3:väskan 4:på 5:bordet>
PATTERN = p[S;< 1, 2, <>, <3, 6>, <>>]
FEATS:MOOD = decl
FEATS:SUBCAT = <>
DTRS:SUBJ = x[STRING = 1]
DTRS:HEAD = y[vp; PATTERN = p, STRING = <2 3 4 5>]
y:FEATS:SUBCAT = <x>
y:DTRS:HEAD = v[verb; STRING = 2]
v:FEATS:SUBCAT = <x[np], z[np], w[pp]>
y:DTRS:CDTRS:ITEMS = <z[STRING = 3], w[STRING = 6<4 5>]>
w:PATTERN = [PP;<4, <5>>]
w:FEATS:SUBCAT = <u[np]>
w:DTRS:HEAD = [prep; STRING = 4]
w:DTRS:CDTRS = <u[STRING = 5]>

---

[1] x, y, z, ... are variables indicating structure sharing. Numbers 1, 2, 3, ... are also variables but always used for strings or patterns. Type names are written at the very beginning of a node. The types clause, np, vp and pp are all assumed to be subtypes of 'phrase', while v is a subtype of 'word'. The clause frame is assumed to have five positions. Its structure is further explained below.

The grammar as a whole defines the possible grammatical descriptions. In addition to the feature structures representing individual words, the grammar contains rules describing hierarchical and sequential relations and principles applying across rules. Every phrase structure rule expresses a relation between values of the attributes STRING, PATTERN, FEATS and DTRS for a local phrase, comprising a dominating item (a mother) and one or more items that it dominates (the daughters). The string of the unit can actually be computed from the pattern by a simple function. The relation between the string and the pattern of a phrase thus need not be specified for each individual rule. However, if the grammar is supposed to be used by a parser, we need to go in the opposite direction, which is not as simple. There are many patterns that yield the same string; e.g. the patterns <np v e <pp> e>, <np v <pp> e e>, <np v e e pp>, <np v e e <pp>>, where 'e' represents the empty string, all yield the string <np v pp>. Moreover, to filter out hypotheses we also need access to information about features and constituent structure.

For this reason it is probably a good idea to compile the grammar into a form which allows efficient parsing. In the end we would like an automatic compiler, of course, but here I can only illustrate how the topological frames can be taken as the basis for an augmented context-free grammar, using a PATR-style notation. Thus, I will simultaneously develop two sets of rules. The first set, the base grammar, applies to items which are daughters of the same node in phrase structure, while the second set, the string grammar, applies to units which are adjacent in the string.

A string grammar of the chosen format can be parsed in different ways. As will be evident there is a close relationship between the string grammar and ATNs with sub-networks corresponding to positions. Our current implementation, however, uses a bidirectional chart-parser, with a mixed strategy. Predictions are made bottom-up when heads are encountered. From there, parsing continues top-down and inside-out with material appearing to the left of the head being consumed before material to the right. In this way the information associated with the head can be exploited to full advantage. As the parser is still being developed, it is too early to report any results on its behaviour.

Combinatorics

Although the phrase structure rules cannot be stated with the same level of generality as in HPSG, they are far more general than an ordinary phrase structure grammar. Moreover, principles such as the Head Feature Principle and the Subcategorization Principle can still apply.

An assumption we will make is that lexical heads have fixed positions within the frames. In our example grammar the frame for the Swedish main clause will have five positions, where the second position is occupied by a (finite) verb and nothing else. Its structure, with type constraints associated with positions, is displayed in (6).

```
(6) The main clause schema (S):
    <phrase, verb, cluster, cluster, cluster>
```

For ease of reference the positions will be called the foundation (F), the V2-position, the nexus field (N), the complement field (C) and the adverbial field (A), respectively.

For parsing purposes the lexical head is a good predictor for the occurrence of a projection. Given a finite verb it is a good chance that it is part of a main clause. In the string grammar we merge the positions appearing on either side of the lexical head and use (upper case) labels for sequences of clusters, as in (7).

```
(7) String grammar: main clauses (categorial part)
    s  →   F   v   NCA1
```

Here s and v represent strings of the indicated sign types, while F represents the contents of the foundation, and NCA_1 represents the joint contents of the last three positions. We can think of the upper case labels as representing a state of a top-down parser. This state is given by a current position (here indicated by the first letter of the label) and a state associated with parsing that position (indicated by the number attached to the label, if any).

As illustrated in (5), a constituent corresponding to a traditional predicate, is assumed, i.e. a VP consisting of a verb and all of its complement except the subject. This constituent is formed according to the following rule:

```
(8) Base grammar: Finite VPs in main clauses

      vp;
      PATTERN = [S;<1, 2, 3, 4, 5>]
      FEATS:FIN = yes
      DTRS:HEAD = w[verb;FEATS:SUBCAT=cons(x,t),STRING=2]
      DTRS:CDTRS = u[cluster; ITEMS = t, STRING = 4]
```

The rule should be interpreted basically in the same way as an HPSG grammar rule, it states one way in which a phrase can be formed, in this case one option for the expression of finite VPs in Swedish, with the lexical head linked to the V2-position and the complements linked to the C-position. Thus, the relation between phrase structure and topology is accounted for by a specific mapping between the daughters of the phrase and the positions of the frame.

The relation between phrase structure and subcategorization information follows the Subcategorization Principle (Pollard & Sag, 1987: 71). If a verb is subcategorized for a subject, an object and a prepositional object, as the verb *lägga* (put), we can augment (8) with

```
w:FEATS:SUBCAT = <x:np[nom], y:np[obj], z:pp>
```

Then, the Subcategorization Principle accounts for the following additional information to unify with (8):

```
FEATS:SUBCAT: = <x>
u:ITEMS = <y, z>
```

When we look at this rule from the point of view of the string grammar, we see that it involves non-adjacent positions. The part of the rule concerned with the V2-position is already covered by (7), but the role of the verb and the complement position must also be accounted for. Moreover, we need to do this in a way that ensures that the dependencies between verb and complements are maintained. To accomplish this we first extend (7) with some equations:

```
(7')   String grammar: main clauses
         s   →    F  v  NCA1
         1:SOURCE = 3:SOURCE = 0
         0:DTRS:HEAD:DTRS:HEAD = 2
```

The first pair of equations links the cluster categories to the clause via the attribute SOURCE. Through the third equation they are also linked to the head. The third equation states that the lexical head is two levels below its resulting projection. This is not necessarily always the case, but we make this simplifying assumption here.

The source will be inherited by all other concerned cluster categories. For instance we have a rule admitting an empty nexus position:

```
(9) String grammar: Empty nexus rule
      NCA1 → CA1
      0:SOURCE = 1:SOURCE
```

For clusters having complements as initial parts, we will have rules of the following form:

```
(10)   String grammar; Complements in main clauses[1]
         CAi  →  xp CAj
         0:SOURCE = 2:SOURCE
         0:SOURCE:DTRS:PRED:DTRS:CDTRS:ITEMS > 1

         CAi  → xp A_1
         0:SOURCE = 2:SOURCE
         0:SOURCE:DTRS:PRED:DTRS:CDTRS:ITEMS > 1
```

These rules are actually schemas that cover a number of rules which together describe the possibilities for complementation in the language. They should be interpreted as follows: in position C of the clause schema, in state i, a category xp is possible, provided no more complements follow, or only complements allowed in state j of position C. The exact number of rules will depend on how we use the LP-rules. If the LP-rules are taken as a separate component of the string grammar, there will be a relatively small number of rules, but if we want the string grammar to respect the LP-constraints we can encode their effect in the states of the cluster categories.

When a finite VP combines with a subject a complete clause is generated. The position of the subject depends on the type of clause. In the case of unmarked declarative clauses (and the corresponding wh-clauses) it is placed in the first position, while in other clauses, including interrogatives and topicalized clauses, it is placed in the third position.

```
(11)   Base grammar: Subjects in unmarked main clauses

         main-clause;
         PATTERN = [S; <1, 2, 3, 4, 5>]
         FEATS:MOOD = unm
         DTRS:HEAD = [vp; SUBCAT = <x>]
         DTRS:SUBJ:STRING = 1

(12)   Base grammar: Subjects in inverted main clauses

         main-clause;
         PATTERN =  [S; <1, 2, 3, 4, 5>]
         FEATS:MOOD = inv
         DTRS:HEAD = [vp; SUBCAT = <x>]
         DTRS:SUBJ:STRING < 3
```

In (11) the subject string is identified with the string of the first position, as it is a unary position. In (12) on the other hand, the subject is merely included among the elements forming the third position cluster and its sequential order will be determined by LP-rules.

For the application of these rules a language-specific principle is supposed to be at work, the Frame Unification Principle, which says that

[1] The symbols '<' and '>' indicate membership of a list.

a non-maximal projection must share its topological frame (and hence basic rules for linearization) with a maximal projection.[1]

```
(13)   The Frame Unification Principle

          [phrase; DTRS = [headed-phrase;]] ⇒
          [PATTERN = DTRS:HEAD:PATTERN]
```

Thus, the complement rule (7) and the subject rules combine to fill one and the same schema with orthographic material.

The corresponding rules of the string grammar are as in (14) and (15):

```
(14)   String grammar; Subject in unmarked clauses.
         F  →  np
         0:SOURCE:DTRS:SUBJ = 1
         0:SOURCE:FEATS:MOOD = unm

(15)   String grammar; Subject in inverted clauses.
         NCAi   →   np NCAj
         0:SOURCE:DTRS:SUBJ = 1
         0:SOURCE:FEATS:MOOD = inv
         0:SOURCE = 2:SOURCE

         NCAi   →   np CA1
         0:SOURCE:DTRS:SUBJ = 1
         0:SOURCE:FEATS:MOOD = inv
         0:SOURCE = 2:SOURCE
```

When an adjunct combines with a head it will also end up in some position of the head's topological frame, although from a syntactic/semantic point of view the head often functions as a kind of argument to the adjunct. The following rule gives one account of the placement of sentence adverbs in Swedish. (Many other solutions are of course possible.)

```
(16)   Base grammar: Sentence adverbs

          main-clause;
          PATTERN = [S;<1, 2, 3, 4, 5>]
          DTRS:HEAD = h[main-clause;]
          DTRS:ADTRS:ITEMS > x[sadv; STRING < 3]
```

There are similar rules placing adjuncts in the first and fifth positions of a main clause.

---

[1]In addition to unification of complete frames there is also the possibility of unifying positions of two frames with one another. There seems to be little use for this in a Swedish grammar, but for the scrambling phenomena of German, it could turn out to be useful. In these sentences, all complements of verbs in a chain of verbs dominating each other turn up in the same position, the Mittelfeld.

As for the string grammar we have the following corresponding rules, saying that a sentence adverb can be accepted in any state associated with the nexus position and be followed by anything accepted in that state, including nothing.

```
(17)  String grammar; Sentence adverbs.
        NCAi  → sa NCAi
        0:SOURCE:DTRS:ADTRS:ITEMS > 1
        0:SOURCE = 2:SOURCE

        NCAi  → sa CA1
        0:SOURCE:ADTRS:ITEMS > 1
        0:SOURCE = 2:SOURCE
```

References

Ahrenberg, Lars. 1990. *A Grammar Combining Phrase Structure and Field Structure.* PROCEEDINGS OF COLING-90, VOL. 2: 1-6.

Diderichsen, Paul. 1962. *Elementær Dansk Grammatik.* Third Edition, Gyldendal, København.

Engelkamp, Judith, Gregor Erbach and Hans Uszkoreit. 1992. *Handling Linear Precedence Constraints by Unification.* PROCEEDINGS OF THE 3OTH ANNUAL MEETING OF THE ACL.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum and Ivan Sag. 1985. *Generalized Phrase Structure Grammar.* Oxford, Basil Blackwell.

Pollard, Carl and Ivan Sag. 1987. *Information-Based Syntax and Semantics.* Volume 1: Fundamentals. CSLI LECTURE NOTES 13, Center for the Study of Language and Information.

Reape, Mike. 1989. *A Logical Treatment of Semi-Free Word Order and Bounded Discontinuous Constituency.* PROCEEDINGS OF THE 4TH MEETING OF THE EUROPEAN CHAPTER OF THE ACL, Manchester, England.