# A Grammar Combining Phrase Structure and Field Structure

Lars Ahrenberg
Department of Computer and Information Science
Linköping University
S-581 83 Linköping, Sweden
email: lah@ida.liu.se

## Abstract

A grammar formalism, Field and Category Grammar (FCG), is described, which beside constituent structure and functional structure recognizes a level of field structure. It is argued that the formalism offers descriptive and computational advantages for the analysis of Scandinavian and other languages that distinguish clause types topologically. It is also argued that the clear distinction between fields and constituents makes FCGs theoretically sounder than other proposed field grammars. A comparison is made of the word order rules and assumptions of FCGs with the partial orderings employed by GPSG and other models.

## 1 Motivations for field structure

Descriptive grammatical works on Germanic languages often refer to notions such as *field* and *schema*, some early, major works being [Dider46] and [Drach37]. Recently, [Rue87] and [Togeby88] have argued that field grammars in Diderichsen's tradition are useful for the computational analysis of Danish. If they are right, the same is obviously true for the other Scandinavian languages and possibly other Germanic languages as well.

A major motivation for field structure in the Scandinavian languages is the correlation between the position of a constituent and grammatical function. For instance, a NP occuring after the finite verb but before the sentence adverbs, i.e. in the field that Diderichsen termed *Neksusfelt*, is a subject, while NPs appearing after the sentence adverbs, in the *Indholdsfelt* (content field) are objects. As main clauses have no surface VP-node consisting solely of the verb and its complements, a configurational definition of subjects and objects is less appealing.

There is a correlation also between positions and thematic functions, the classical example being Diderichsen's *Fundament* (foundation), the position before the finite verb which holds thematically prominent constituents of various kinds.

A second motivation is that the word order regularities of clause types are better described if we have access to field structure. In a phrase structure grammar we either have to write a separate rule, or rule schema, for each clause type, or else introduce powerful rules such as transformations or meta-rules to capture differences and similarities. Field structure can be used to express the common traits directly: the schema in figure 1 apply to virtually all Swedish clause types.[1] Moreover, variation can be accounted for in terms of constraints on what may occur in the fields and such constraints may be expressed by regular expressions. Thus, the incorporation of field structure to a formalism does not add to its computational complexity.

## 2 Field structure vs. phrase structure

It is obvious that schemas such as that of figure 1 can be defined by context-free rewrite rules each of which specifies a number of subfield-relations and a sequential order for the subfields. The rules below together define the schema in figure 1, which we name $\Sigma$.

(1) $\Sigma \rightarrow$ F NexF ContF
NexF $\rightarrow$ v nex
ContF $\rightarrow$ v' ObjF adv
ObjF $\rightarrow$ obj pobj comp

The simplest way of formalizing a field grammar is to define an appropriate set of rules of this kind and, if we want to derive a functional structure, associate the rules and lexical entries with functional information. This is essentially the approach taken by [Rue87] and by [Togeby88]. As a result the field notion is merged with the notion of constituent. It is indeed often said that an advantage of Diderichsen's analysis is that it offers a better constituent analysis of Danish than, say, traditional TG. This is not so, however. On the contrary, it is one of the weaknesses of Diderichsen's work that the notions of fields and constituents are regularly confused (cf.

[1] The schema in figure 1 is a revised version of Diderichsen's classical schema. For instance, the nexus field has been given two positions instead of three for reasons explained in section 3.1.3.

| Foundation (F) | Nexus field (NexF) | | Content field (ContF) | | | | |
|---|---|---|---|---|---|---|---|
| | v | nex | v' | ObjF | | | adv |
| | | | | obj | pobj | comp | |
| Jag | hann | inte | — | — | — | träffa honom | idag |
| *I* | *managed* | *not* | — | — | — | *see him* | *today* |
| — | att | jag inte | hann | — | — | träffa honom | idag |
| — | *that* | *I not* | *managed* | — | — | *see him* | *today* |

Figure 1: A schema for the Swedish clause with two analysed examples.

[Telem72,Braunm86]). Instead field structure is better conceived of as a level that accounts for the linearization of independently defined constituents.

While such a conception of field structure is more restricted, it is more motivated and equally amenable to formalization. The formalism must deal with two types of information for a constituent, however, its category and the field(s) it occurs in. Also, we need to distinguish carefully the dominance relations for fields (*superfield*) and categories (*dominates*) as they differ in their logical properties. Here only two important differences will be noted: [1] Fields *transmit* expressions, categories don't. Given an expression, e, that is situated in a field, f, it will also be situated in every field that is a superfield of f. Conversely, fields generally allow multiple occurrences of constituents (incl. none; cf. figure 1), while categories categorize exactly one constituent at a time. [2] The *superfield*-relation is non-recursive, which means that schemas have a finite number of elements. The *dominates*-relation, on the other hand, allows recursion in the usual way.

## 3 Field-and-Category Grammars

Field-and-Category Grammars (henceforth FCG) may, with functional schemas included, be regarded as a generalization of Lexical-Functional Grammars [LFG82]. There is then the usual division between two structural levels, but as the surface level includes information about the position of a constituent in a relevant schema, and not just category, we refer to it as a *topological structure*, or *t-structure*. For the purposes of this paper the f-structure may be taken as in LFG.

A t-structure for a simple sentence is illustrated in figure 2. The rules necessary for the generation of t-structures form a *Basic FCG*.

A *schema* is defined as a maximal field. A *position* is a terminal field. An *identifier position* is a position that admits an identifier of a phrase, such as a lexical head.

Categories are ordered by a subsumption relation. An *actual category* is a category that does not subsume any other category; an *abstract category* is one

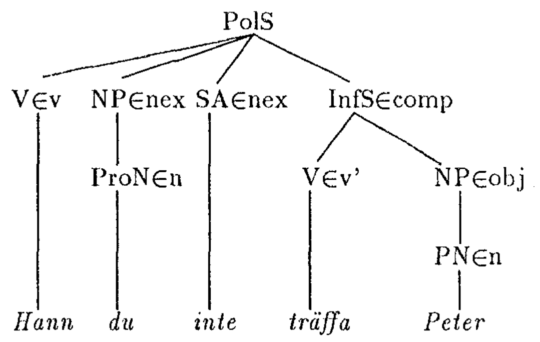

Figure 2: A topological structure for the sentence *Hann du inte träffa Peter?* (Didn't you manage to see Peter?). Nodes are labelled C∈p, where C indicates the category and p the position of the dominated string.

that does. Abstract categories express what is common to a class of actual categories.

A *configuration* is a triple [D, C, p] where D is an actual category, C is an actual category or a word, and p is a position. A configuration corresponds to a branch of a local tree of a t-structure. D is the category of the dominating node, C the category of a dominated node and p the position of the latter in the schema associated with the former. Conversely, a local tree can be represented as a multiset of configurations that have a common first element. For instance, the top local tree of figure 2 corresponds to the set {[PolS, V, v], [PolS, NP, nex], [PolS, SA, nex], [PolS, InfS, comp]}. Multisets are required in the general case as there may be several daughters with the same category and position.

### 3.1 Rule types

#### 3.1.1 Field structure rules

Field structure rules define the internal structure of fields in terms of subfields. In addition, they assign each position an occurrence index, stating the maximum number of fillers it takes. I will write $p^*$ to indicate a position with any number of fillers, $p^n$ for a position with a maximum of n fillers, (p) for a position with one optional filler, and simply p for a position with one obligatory filler. The rules in (1) may be expanded as below, where a simplified rule for the noun phrase schema is also stated.

| Notation | Meaning |
|---|---|
| x∈p | p must not be empty |
| e∈p | p must be empty |
| A∈p | p must contain an A |
| w∈p | p must contain word w |
| (A)∈p | p may only contain an A |
| (w)∈p | p may only contain word w |
| A˜∈p | p must not contain an A |

Table 1: Basic topological constraints of a FCG.

(2) Σ → (F) NexF ContF
NexF → (v) nex*
ContF → (v') ObjF adv*
ObjF → $obj^2$ pobj* (comp)

(3) Π → (det) mod* n rel*

### 3.1.2 Category definitions

Category definitions define necessary properties of categories. They may be written as 4-tuples (C, C', T, F) where C is defined as a subcategory of C' meeting the topological constraints T, and the functional constraints F.

Basic topological constraints state what must or may occur in a specific position. A list of basic topological constraints is found in table 1. The element T is a conjunction of such basic constraints, or a schema symbol. In the latter case the definition includes a *category-schema association*, which says that the category, and, by implication, all its subcategories, are linearized by the named schema. The other constraints give information about what occurs in specific positions of that schema.

The functional constraints are written as conjunctions of attribute-value assignments and value constraints. A single attribute name indicates that this attribute must have a value at f-structure.

Some examples of category definitions are given below. Together they define an inheritance hierarchy of constituent categories, where properties of a category high up in the hierarchy are shared by categories below it. Topological properties that are common to a set of actual categories are expressed at their common ancestors, and, in particular, by the common schema they inherit.

csa1: (S, –, Σ, Subj∧Pred∧Vform)
csa2: (NP, –, Π, Numb∧Gend)

def1: (MainS, S, V∈v, –)
def2: (V2S, MainS, x∈F, –)
def3: (V1S, MainS, e∈F, –)
def4: (PolS, V1S, NP∈nex, Vform=Fin)
def5: (ImpS, V1S, NP˜∈nex, Vform=Imp)
def6: (SubS, S, (Comp)∈v∧V∈v', –)
def7: (InfS, SubS, e∈F∧(*att*)∈v ∧NP˜∈nex, Vform=Inf)

For instance, in (def4) a polar interrogative clauses (PolS) is defined as a verb-first clause (V1S), which in (def3) is defined as a main clause (MainS), which in turn is defined as a clause (S). Being a clause it is linearized by Σ according to (csa1) and its f-structure must have a subject, a semantic form and a verbal property. Being a main clause it has a verb in position v (def1). Being a verb-first clause it has an empty foundation. In distinction to other verb-first clauses it has a finite verb form, and an expressed subject in position nex.

### 3.1.3 Configuration rules

While category definitions state what must hold of a given category, configuration rules state what may hold of any category. Each configuration of the language is defined by some configuration rule. A configuration rule may be written as a list of the form (CS, F, i) where CS is a description of a set of configurations, F is a conjunction of functional constraints and i is an occurrence index. We take advantage of the category hierarchy and use abstract categories in the description of configuration sets. Three illustrations are given below:

conf1: ([S, NP, F], ↑SUBJ=↓, 1)
conf2: ([S, NP, nex], ↑SUBJ=↓, 1)
conf3: ([S, SA, nex], ↑=↓, *)

The arrows, ↑ and ↓ are used as in LFG: The up-arrow identifies the f-structure of the dominating node of the configuration, whereas the down-arrow identifies the f-structure of the dominated node.

The first two rules state the possible subject configurations of Swedish. They apply to all subcategories S and NP, unless this is contradicting the definitions of these categories. For instance, (conf1) does not apply to a V1S as defined in (def3).

The last two rules both define fillers of position 'nex' without ordering them. The third rule defines an iterative configuration, as indicated by its occurrence index. Thus, the subject is allowed to take different relative positions w r t the sentence adverbs in agreement with the facts illustrated in (4)-(6). In this way fields serve to define borders for local word order variation.

(4) I natt var katten nog inte ute.
*last-night was the-cat* nog *not out*
"Probably, the cat wasn't outdoors last night"

(5) I natt var nog katten inte ute
*last-night was* nog *the-cat not out*

(6) I natt var nog inte katten ute.
*last-night was* nog *not the-cat out*

### 3.1.4 Lexical rules

A lexical rule may be written on the form (w, C, T, F) where the lexical item w is assigned a category, a (usually null) topological constraint and some functional information. Three illustrations are given in (7)-(9).

(7) (hann, V, —, Pred='manage<(Subj)(Xcomp)>'
∧Xcomp:Subj=Subj
∧Vform=Fin∧Tense=Pret)
(8) (inte, SA, —, Pol=Neg)
(9) (Peter, N, —, Pred='Peter'
∧Numb=Sg∧Gend=Utr)

### 3.1.5 Well-formedness conditions

In order to be well-formed an expression of a FCG must have both a well-formed t-structure and a well-formed f-structure. We omit the requirements of well-formed f-structures as they can be taken to coincide with those of a LFG.

A topological structure, T, is well-formed according to a FCG, G, iff the following condition holds: (i) Each node of T is assigned an actual category and every node apart from the top-node is assigned a position; (ii) Any local tree, L, of T, with top-node category, C, satisfies the following conditions: (a) for each branch of L there is a configuration rule, or a lexical rule, in G that licenses it; (b) if C is non-terminal, there is a schema, $\sigma$, associated with C, such that the sequential order of the branches is in agreement with the sequential order of their positions in $\sigma$; (c) all restrictions on $\sigma$ imposed by C in its definition are satisfied by L.

## 4 Properties of Basic FCGs

By removing all functional information from a FCG we obtain a Basic FCG. It is the Basic FCG that is responsible for the expression of dominance and precedence relations in the grammar, i.e. it has the same role as the phrase-structure rules of a LFG. This section is concerned with some interesting properties of Basic FCGs. First I show that a Basic FCG is weakly equivalent to a context-free grammar.

Let G be a Basic FCG. Let A be the set of actual categories, Z the set of schemas, and P the set of positions, all finite sets. For any C∈A let L(C) denote the set of strings dominated by C. The language of G, L(G) is defined as a union of such sets for some suitable subset A' ⊂ A, e.g. by the set of subcategories of S.

Let W be the set of words that occur in configuration rules and category definitions. Let K be the set A∪W.

For any $\sigma$ ∈S we may, by expansion of the relevant field structure rules, derive a *positional structure* for $\sigma$. Call this structure $e_\sigma$. For instance, from (2) we may derive a positional structure $e_\Sigma$:

(F) (v) nex* (v') obj$^2$ pobj* (comp) adv*

A positional structure can be given the form of a regular expression over P. This is guaranteed, since fields are non-recursive objects.

Let D be any actual category that is linearized by $\sigma$, and let p be a position that occurs in $e_\sigma$. The category definitions associate with D and p a conjunction of topological conditions, $D_{p,\tau}$, where each conjunct has one of the forms in table 1.

For given D and p the configuration rules allow us to derive the constituent strings that may occur in p under D. There is only a finite number of applicable configuration rules. Each rule gives a disjunction of actual categories and an occurrence index for that disjunction. If all occurrence indices are finite, or if the occurrence index of p is finite, the constituent strings may be represented by a finite language over K. If some occurrence indices are '*', and p itself has occurrence index '*', we may first form a finite sublanguage over K that represents all strings of non-iterative constituent categories, and then extend it by introducing the iterative constituents. In either case, the result is a regular language over K. We call this language $L_{D,p}$.

For instance, assuming that (conf1) and (conf2) are the only rules pertaining to position nex, and that NP has three actual subcategories, CNP, PNP and ProNP, we have $L_{PolS,nex} = L_{S,nex}$ = SA*(CNP + PNP + ProNP)SA*.

Given $L_{D,p}$ we want to derive the sublanguage of constituent strings that satisfy $D_{p,\tau}$. Call this language $L_{D,p,\tau}$. Consider first the primitive cases:

1. If $D_{p,\tau}$ = e∈p then $L_{D,p,\tau}$ = {e}.
2. If $D_{p,\tau}$ = x∈p then $L_{D,p,\tau}$ = $L_{D,p}$–{e}.
3. If $D_{p,\tau}$ = A∈p where A is actual,
then $L_{D,p,\tau}$ = $L_{D,p}$∩K*AK*.
4. If $D_{p,\tau}$ = A∈p where A is abstract,
then $L_{D,p,\tau}$ = $L_{D,p}$∩(K*$A_1$K*∪···∪K*$A_n$K*)
where $A_1$, ..., $A_n$ are actual subcategories of A.
5. If $D_{p,\tau}$ = (A)∈p where A is actual,
then $L_{D,p,\tau}$ = $L_{D,p}$∩(K*AK*∪{e}).
6. If $D_{p,\tau}$ = (A)∈p where A is abstract,
then $L_{D,p,\tau}$ = $L_{D,p}$∩(K*$A_1$K*∪···∪K*$A_n$K*∪{e}),
where $A_1$, ..., $A_n$ are actual subcategories of A.
7. If $D_{p,\tau}$ = $A^{\sim}$∈p then $L_{D,p,\tau}$ =
$L_{D,p}$∩(K*–K*AK*)
8. If $D_{p,\tau}$ = w∈p then $L_{D,p,\tau}$ = $L_{D,p}$∩K*wK*.
9. If $D_{p,\tau}$ = (w)∈p
then $L_{D,p,\tau}$ = $L_{D,p}$∩(K*wK*∪{e}).

In all cases $L_{D,p,\tau}$ is a regular set. As $D_{p,\tau}$ in the general case is a conjunction of such primitive constraints, it follows that $L_{D,p,\tau}$ will always be a regular set over K.

Let $L_D$ be the totality of constituent strings that D may dominate. Then $L_D$ is obtained by substitution of $L_{D,p,\tau}$ for p in $e_\sigma$. As the class of regular sets is closed under substitution, $L_D$ will also be a regular set over K. As D itself may occur in $L_D$, we may have recursive categories in L(D), however. In any case, L(D), and by implication, L(G), is a context-free language.

It is interesting to note that many simple context-free languages cannot be given a simple Basic FCG. For example, if a certain category, C, takes one obligatory daughter, H, and two optional daughters A, B, according to the the CF-grammar G1, there is no Basic FCG for L(G1) that has C as an actual category.

(G1) $C \rightarrow H$
$C \rightarrow H\ A$
$C \rightarrow B\ H$
$C \rightarrow A\ H\ B$

If there is such a FCG, it must employ at least three positions, since otherwise alternative orders must be allowed. Thus it takes three configuration rules pertaining to three different positions to account for the string [A H B]. But as these are independent the strings [A H] and [H B] can also be generated, contradicting the assumption.

In a Basic FCG a category behaving as C in G1 must be abstract and its different realizations must be divided among a number of actual subcategories. A Basic FCG weakly equivalent to G1 is G2:

(G2) (fsr1) $\sigma \rightarrow$ (p1) p2 (p3)

(csa1) $(C, -, \sigma, -)$
(cdef1) $(C, -, H \in p2, -)$
(cdef2) $(C1, C, e \in p1 \wedge (A) \in p3, -)$
(cdef3) $(C2, C, B \in p1 \wedge e \in p3, -)$
(cdef4) $(C3, C, A \in p1 \wedge B \in p3, -)$

(conf1) $([C, H, p2], -, 1)$
(conf2) $([C, A, p1], -, 1)$
(conf3) $([C, B, p1], -, 1)$
(conf4) $([C, A, p3], -, 1)$
(conf5) $([C, B, p3], -, 1)$

What languages can FCGs describe well? Intuitively it seems that complex constituents that share a set of potential daughters should obey the same constraints as regards their relative order and occurrence. In particular, the occurrence of one daughter should be independent of the occurrence of other daughters. Where there is a difference in these properties, there must be a categorial distinction in the grammar, as the example above illustrates. We may call this property *category-dependent fixed ordering*. It seems, however, that this property is significant for natural languages, at least for those, like the Germanic languages, that distinguish clause types on topological grounds.

## 5 Field structure and partial orderings

If the (surface) syntactic structures of a natural language are specified by means of a context-free grammar as in LFG, there is no chance of expressing any generalizations pertaining to word order. LFG admits a number of notational devices to facilitate the writing of c-structure rules, but has made few claims about possible word order restrictions. [GPSG85], on the other hand, makes the strong claim that natural languages obey Exhaustive Constant Partial Ordering (ECPO), i.e. that the possible linearizations of a set of sister constituents are the same in any local tree irrespective of the categories of the mother and other sisters. Such linearizations are expressed by means of partial orderings, or LP-rules, of the form $A \prec B$.

It is obvious that this assumption is more naturally made in a framework that works with local trees that have only two or three branches than in a framework which employs flat structures. For instance, the existence of unmarked and inverted clauses is not contradicting the ECPO-hypothesis, if the subject is regarded as a sister of the finite verb only in the inverted case. However, there are constructions that speak against it as a universal, such as the order of object and verb in German main and subordinate clauses: *Ich kaufte ein Auto* (I bought a car) vs. *Ich habe ein Auto gekauft* (I have a car bought = I have bought a car), and the order of verb participles and their complements in Swedish predicative and attributive constructions: *Rapporten är beställd av Borg* (The report is ordered by Borg) vs. *Den av Borg beställda rapporten* (The by Borg ordered report = The report that Borg ordered). These constructions are not problematic for FCGs, however, although they necessitate a categorial split.

Although the number of categorial splits can be many in a FCG, one would not like the number of schemas to be very high. For a language like Swedish it seems possible to limit the description to five schemas, one for each type of projection (V, N, A, P) and one for coordinated structures [Ahrenb89].

LP-rules are used also in frameworks which do not subscribe to the ECPO-property, such as HPSG [PolSag87]. However, they need to be complemented by something, as they miss an important aspect of word order. As they apply to sister constituents, they fail to give any information on the position of a daughter relative to the phonological span of the mother. For instance, as a speaker of English I know that the definite article appears at the very beginning of an NP and that relative clauses appear at the end. Given a set of LP-rules ordering determiners, relative clauses and other NP-constituents we may possibly infer this information, but this is a roundabout way of doing it. To express such facts directly we need a device that will impose a sequential structure on phonological spans, and it is for this purpose that the topological schema is useful.

On the other hand partial orderings seem better suited to describe category-independent word order regularities. Consider the case of complements to a head. In the Germanic languages the normal order would be the one expressed in (10): NP-complements precede PP-complements which precede verbal complements whatever the category of the head [GPSG85, p. 110].

(10) $\mathrm{NP} \prec \mathrm{PP} \prec \mathrm{VP}$

The rule in (2) defining the complement field (ObjF), repeated here for convenience, specifies three positions, one for bare objects, one for prepositional objects and one for verbal and adjectival complements.

$\mathrm{ObjF} \rightarrow \mathrm{obj}^2\ \mathrm{pobj}^*\ (\mathrm{comp})$

Even if we could appeal to the same or a similar field structure rule in the case of complements to the adjective, it seems natural in this case to explain the ordering in terms of the difference in category between different complements. Thus, with the introduction of (10) ObjF could be regarded as a position, i.e. as a terminal of the schema in figure 1.

Note however that in a FCG LP-rules receive a slightly different interpretation. They apply to positions rather than to local trees.

## 6 Concluding remarks

Current work on FCG includes the implementation of a head-driven, bidirectional chart-parsing algorithm. The basic idea is to use fillers of identifier positions to trigger bottom-up predictions. FCGs have the advantage that the search for topologically different, alternative projections of a head or other identifier, can be accomplished by a single active edge. On the other hand the category of an edge is often abstract, and has to be determined on the basis of category definitions and the content of the edges that combined to introduce it.

Finally it should be stressed that while FCG is a variant of a LFG, the idea of regarding the schemas of traditional field grammars as structures of partial information can of course be exploited in any unification-based formalism.

## References

[Ahrenb89] L. Ahrenberg: A formal field grammar. Research report LiTH-IDA-89-46, Linköping university, department of Computer Science.

[Bresn82] J. Bresnan (ed.): *The Mental Representation of Grammatical Relations*, The MIT Press: Cambridge Mass.

[Braunm86] K. Braunmüller: Hvor moderne er P. Diderichsens sætningsanalyse? I [H&A 86] pp. 77-98.

[Dider46] P. Diderichsen: *Elementær Dansk Grammatik.* Third edition. Copenhagen, Gyldendal.

[Drach37] E. Drach: *Grundgedanken der Deutschen Satzlehre.* Frankfurt/M, Diesterweg; reprint Darmstadt, Wiss. Buchgesellschaft, 1963.

[GPSG85] G. Gazdar, E. Klein, G. Pullum and I. Sag: *Generalized Phrase Structure Grammar.* Oxford, Basil Blackwell, 1985.

[H&A 86] L. Heltoft and J. E. Andersson (eds.): *Sætningsskemaet og dets stilling — 50 år efter. Nydanske studier & almen kommunikationsteori 16-17.* Akademisk Forlag, 1986.

[LFG82] R. M. Kaplan and J. Bresnan: Lexical-Functional Grammar: A Formal System for Grammatical Representation. In [Bresn82], pp. 173-281.

[PolSag87] C. Pollard and I. A. Sag: *Information-Based Syntax and Semantics. Volume 1: Fundamentals.* CSLI Lecture Notes, No. 13. Stanford.

[Rue87] H. Rue: Danish field grammar in typed PROLOG. *Proceedings of the Third Conference of the European Chapter of the ACL*, Copenhagen, April 1-3, 1987: 167-172.

[Telem72] U. Teleman: Om Paul Diderichsens syntaktiska modell. I Teleman, Ulf, *Tre uppsatser om grammatik.* Studentlitteratur, Lund, 33-57.

[Togeby88] O. Togeby: Parsing Danish Text in Eurotra. *Nordic Journal of Linguistics*, Vol. 11, Nos. 1-2, p. 175-191.