Two Useful Measures of Word Order Complexity

Tomáš Holan; Vladislav Kuboň; Karel Oliva; Martin Plátek

Abstract
This paper presents a class of dependency-based formal grammars (FODG) which can be parametrized by two different but similar measures of non-projectivity. The measures allow to formulate constraints on the degree of word-order freedom in a language described by a FODG. We discuss the problem of the degree of word-order freedom which should be allowed by a FODG describing the (surface) syntax of Czech.

1 Introduction
In [Kubon,Holan,Plátek,1997] we have introduced a class of formal grammars. Robust Free-Order Dependency Grammars (RFODG’s), in order to provide for a formal foundation to the way we are developing a grammar-checker for Czech, a natural language with a considerable level of word-order freedom. The design of RFODG’s was inspired by the commutative CF-grammars (see [Huynh,83]), and several types of dependency based grammars (cf., e.g., [Gaifman,1965]. [Béleckij,1967], [Plátek,1974], [Nečuk,1988]). Also in [Kubon,Holan,Plátek,1997], we have introduced different measures of incorrectness and of non-projectivity of a sentence. The measures of the non-projectivity create the focus of our interest in this paper. They are considered as the measures of word-order freedom. Considering this aim we work here with a bit simplified version of RFODG’s, namely with Free-Order Dependency Grammars (FODG’s). The measures of word-order freedom are used to formulate constraints which can be imposed on FODG’s globally, or on their individual rules.

Two types of syntactic structures, namely DR-trees (delete-rewrite-trees), and De-trees (dependency trees), are connected with FODG’s. Any DR-tree can be transformed into a De-tree in an easy and uniform way. In [Kubon,Holan,Plátek,1997] the measures of non-projectivity are introduced with the help of DR-trees only. Here we discuss one of them, called node-gaps complexity (Ng). It has some very interesting properties. CFL’s are characterized by the complexity 0 of Ng. The Ng also characterizes the time complexity of the parser used by the above-mentioned grammar-checker. The sets of sentences with the Ng less than a fixed constant are parsable in a polynomial time. This led us to the idea to look for a fixed upper bound of Ng for all Czech sentences with a correct word order. In [Kubon,Holan,Plátek,1997] we even worked with the conjecture that such an upper bound can be set to 1. We will show in Section 5 that it is theoretically impossible to find such an upper bound, and that even for practical purposes, e.g., for grammar-checking, it should be set to a value considerably higher than 1. This is shown with the help of the measure dNg which is introduced in the same way as Ng, but on the dependency trees. dNg creates the lower estimation for Ng. The advantage of dNg is that it is linguistically transparent. It allows for an easy discussion of the complexity of individual examples (e.g., Czech sentences). On the other hand, it allows neither for characterizing the class of CFL’s, nor for imposing the context-free interpretation for some individual rules of a FODG. Also, no useful relation between dNg and some upper estimation of the time complexity of parsing has been established yet. These complementary properties of Ng and dNg force us to consider both of them simultaneously here.

2 FOD-Grammars
The basic notion we work with are free-order dependency grammars (FODG’s). In the sequel the FODG’s are analytic (recognition) grammars.

Definition -(FODG). First-order dependency grammar (FODG) is a tuple \( G = (T, N, S, P) \).
where the union of $N$ and $T$ is denoted as $V$, $T$ is the set of terminals, $N$ is the set of nonterminals, $S_r \subseteq V$ is the set of root-symbols (starting symbols), and $P$ is the set of rewriting rules of two types of the form:

a) $A \rightarrow_X BC$, where $A \in N$, $B, C \in V$, $X$ is denoted as the subscript of the rule, $X \in \{L, R\}$.

b) $A \rightarrow B$, where $A \in N$, $B \in V$.

The letters $L$ ($R$) in the subscripts of the rules mean that the first (second) symbol on the right-hand side of the rule is considered dominant, and the other dependent.

If a rule has only one symbol on its right-hand side, we consider the symbol to be dominant.

A rule is applied (for a reduction) in the following way: The dependent symbol is deleted (if there is one on the right-hand side of the rule), and the dominant one is rewritten (replaced) by the symbol standing on the left-hand side of the rule.

The rules $A \rightarrow_L BC$, $A \rightarrow_R BC$, can be applied for a reduction of a string $z$ to any of the occurrences of symbols $B,C$ in $z$, where $B$ precedes (not necessarily immediately) $C$ in $z$.

For the sake of the following explanations it is necessary to introduce a notion of a $DR$-tree (delete-rewrite-tree) according to $G$. A $DR$-tree maps the essential part of history of deleting dependent symbols and rewriting dominant symbols, performed by the rules applied.

Put informally, a $DR$-tree (created by a FODG $G$) is a finite tree with a root and with the following two types of edges:

a) **vertical**: these edges correspond to the rewriting of the dominant symbol by the symbol which is on the left-hand side of the rule (of $G$) used. The vertical edge leads (is oriented) from the node containing the original dominant symbol to the node containing the symbol from the left-hand side of the rule used.

b) **oblique**: these edges correspond to the deletion of a dependent symbol. Any such edge is oriented from the node with the dependent deleted symbol to the node containing the symbol from the left-hand side of the rule used.

We will say that a $DR$-tree $Tr$ is complete if for any of its leaves $U = [A, i, j, e]$, where $i > 1$, it holds that there is exactly one leaf with the horizontal index $i - 1$ in $Tr$.

**Example 1.** This formal example illustrates the notion of $FODG$. The following grammars $G_1$, $G_2$ are $FODG$'s. $G_1 = (N_1, T_1, \{S\}, P_1)$. $T_1 = \{a, b, c\}$, $N_1 = \{T, S\}$, $P_1 = \{S \rightarrow_{L} aT\}, S \rightarrow_{R} T a. T \rightarrow_{L} bc, T \rightarrow_{R} cb\}$.

$G_2 = (N_2, T_2, \{S\}, P_2)$. $T_2 = \{a, b, c, d\}$, $N_1 = \{S, S_1, S_2\}$, $P_2 = \{S \rightarrow_{R} S_1 a, d, S_1 \rightarrow_{R} S_2 b, S_2 \rightarrow_{R} S e\}.$

Fig. 1 displays a $DR$-tree generated by $G_1$ for the input sentence $aabbcc$. 

**Definitions.** $TX(G)$ denotes the set of complete $DR$-trees rooted in a symbol from $S_r$ created by $G$. If $Tr \in TX(G)$, we say that $Tr$ is *parse*d by $G$.

22
I

Figure 1: A DR-tree Tr generated by the grammar GI. The nodes of Tr are L1 = [a, 1, 1.1], L2 = [a, 2, 1.2], L3 = [b, 3, 1.3], L4 = [b, 4, 1.4], L5 = [c, 5, 1.5], L6 = [c, 6, 1.6] (the leaves), and N1 = [T, 1.1, 1], N2 = [T, 2, 2.1], N3 = [S, 2, 2, 1.2], N4 = [S, 1.2, 1.3].

Let w = a1a2...an, u' ∈ TN(G), and let [ai, i, c(i)] denote the i-th leaf of Tr for i = 1,...,n. In such a case we say that the string w is parsed into Tr by G.

The symbol L(G) represents the set of strings (sentences) parsed into some DR-tree from TN(G).

We will also write TN(u', G) = {Tr: u' is parsed into Tr by G}.

Example 2. Let us take the grammar G1 from the example 1. Then L(G1) = {u' ∈ {a, b, c}+|u' contains the same number of a’s, b’s, and c’s}. Let us take the G2. Then L(G2) = {du' | u' ∈ L(G1)}.

L(G1), L(G2) are two variants of a well known non-context-free language.

Let us now introduce dependency trees parsed from a string. Informally, a dependency tree is obtained by contracting each vertical path of a DR-tree into its (starting) leaf.

Definition - (De-tree). Let Tr ∈ TN(w, G) (w is parsed into Tr by G), where w = a1a2...an. The dependency tree dTr(T) contracted from Tr is defined as follows: The set of nodes of dTr(T) is the set of 3-tuples [ai, i, k(i)] (note that ai is the i-th symbol of w). k(i) = 0 if and only if the root of Tr has the horizontal index i (then the [ai, i, k(i)] is also the root of dTr(T)). k(i) ∈ N if and only if in Tr an oblique edge leads from some node with the horizontal index i to some node with the horizontal index k(i).

We can see that the edges of dTr(T) correspond (one to one) to the oblique edges of Tr, and that they are fully represented by the second and the third slot of nodes of dTr(T). The second slot is called horizontal index of the node.

The symbol dTN(u', G) denotes the set of dTr(T) where Tr ∈ TN(w, G). The symbol dTN(G) denotes the union of all dTN(u', G) for u' ∈ L(G). We say that dTN(G) is the set of De-trees parsed by G.

An example of a dependency tree is given in Fig. 2.

3 Discontinuity measures

In this section we introduce two measures of non-projectivity (discontinuity).

First we introduce the notion of a coverage of a node of a DR-tree.

Definition. Let Tr be a DR-tree. Let u be a node of Tr. As Cov(u, Tr) we denote the set of horizontal indices of nodes from which a path (bottom up) leads to u. (Cov(u, Tr) obligatorily contains the horizontal index of u). We say that Cov(u, Tr) is the coverage of u (according to Tr).

Example 3. This example shows the coverage of individual nodes of Tr1 from Fig.1.

Cov([a, 1, 1, 1], Tr1) = {1}.
Cov([a, 2, 1, 2], Tr1) = {2}.
Cov([b, 3, 1, 3], Tr1) = {3}.
Cov([b, 4, 1, 4], Tr1) = {4}.
Cov([c, 5, 1, 3], Tr1) = {5}.
Cov([c, 6, 1, 4], Tr1) = {6}.
\[ \text{Cor}([T, 4, 2, 1], T_{r1}) = \{4, 6\}, \]
\[ \text{Cor}([T, 3, 2, 2], T_{r1}) = \{3, 5\}, \]
\[ \text{Cor}([S, 2, 2, 1], T_{r1}) = \{2, 3, 5\}, \]
\[ \text{Cor}([S, 1, 2, 1], T_{r1}) = \{1, 4, 6\}, \]
\[ \text{Cor}([S, 1, 3, 0], T_{r1}) = \{1, 2, 3, 4, 5, 6\} \]

Let us define a (complexity) measure of non-projectivity by means of the notion of a coverage for each type of trees:

**Definition.** Let \( T_r \) be a DR-tree. Let \( u \) be a node of \( T_r \), \( \text{Cor}(u, T_r) = \{i_1, i_2, \ldots, i_n\} \), and \( i_1 < i_2 \cdots < i_{n-1} < i_n \). We say that the pair \((i_j, i_{j+1})\) forms a gap if \( 1 \leq j < n \), and \( i_{j+1} - i_j > 1 \). The symbol \( \text{Ng}(u, T_r) \) represents the number of gaps in \( \text{Cor}(u, T_r) \). \( \text{Ng}(T_r) \) denotes the maximum from \( \{\text{Ng}(u, T_r); u \in T_r\} \). We say that \( \text{Ng}(T_r) \) is the node-gaps complexity of \( T_r \).

In the same way \( d\text{Ng}(dT_r) \) can be introduced for any dependency tree \( dT_r \).

**Example 4.** We stick to the DR-tree \( T_{r1} \) from previous examples. The following coverages contain gaps:

\[ \text{Cor}([T, 4, 2, 1], T_{r1}) = \{4, 6\}, \]
\[ \text{Cor}([T, 3, 2, 2], T_{r1}) = \{3, 5\}, \]
\[ \text{Cor}([S, 2, 2, 1], T_{r1}) = \{2, 3, 5\}, \]
\[ \text{Cor}([S, 1, 2, 1], T_{r1}) = \{1, 4, 6\}, \]

has one gap \((4,6)\),

has one gap \((3,5)\),

has one gap \((3,5)\),

has two gaps \((1,4)\) and \((4,6)\).

We can see that \( T_{r1} \) has three different gaps \((1,4),(3,5),(4,6)\), and \( \text{Ng}(T_{r1}) = 2 \).

**Example 5.** This example shows the coverages of the nodes of \( dT_{r1} \) from Figure 2.

\[ \text{Cor}([a, 1, 0], dT_{r1}) = \{1, 2, 3, 4, 5, 6\}, \]
\[ \text{Cor}([a, 2, 1], dT_{r1}) = \{2, 3, 5\}, \]
\[ \text{Cor}([b, 3, 2], dT_{r1}) = \{3, 5\}, \]
\[ \text{Cor}([b, 4, 1], dT_{r1}) = \{4, 6\}, \]
\[ \text{Cor}([c, 5, 3], dT_{r1}) = \{5\}, \]
\[ \text{Cor}([c, 6, 4], dT_{r1}) = \{6\} \]

We can see that \( dT_{r1} \) has two different gaps \((3,5),(4,6)\), and \( d\text{Ng}(dT_{r1}) = 1 \).

**Definitions.** Let \( i \in (\text{Nat} \cup \{0\}) \cup \{\ast\} \) and let \( \ast \) be greater than any natural number. Let us denote as \( TN(w, G, i) \) the set of DR-trees from \( TN(w, G) \) such that the value of the measure \( \text{Ng} \) does not exceed \( i \) on them. When \( i \) is the symbol \( \ast \), it means that no limitation is imposed on the corresponding value of the measure \( \text{Ng} \).

Let us denote \( LN(G, i) = \{w; \ TN(w, G, i) \neq \emptyset\} \). \( TN(G, i) \) denotes the union of all \( TN(w, G, i) \) over all \( w \in L(G, i) \).

\( TN(i) \) denotes the class of sets (of DR-trees) \( TN(G, i) \), for all FOD-grammars \( G \).

\( LN(i) \) denotes the class of languages \( LN(G, i) \), for all FOD-grammars \( G \).

The denotations \( dTN(w, G, i) \), \( dLN(G, i) \), \( dTN(i) \), \( dLN(i) \) can be introduced stepwise in the same way for DR-trees as \( TN(w, G, i) \), \( LN(G, i) \), \( TN(i) \), \( LN(i) \) for DR-trees.

4 Formal observations

This section contains some observations concerning the notions defined above. Due to space limitations, they are not accompanied by (detailed) proofs. The symbol \( CF^+ \) denotes the set of context-free languages without the empty string and \( comCF^+ \) denotes the set of commutative context-free languages without the empty string (\( L \) is a commutative CF-language if it is composed from a CF-language and from all permutations of its words (see, e.g., \( (0) \)) note, in particular, that \( L \) need not be a context-free language, actually the classes \( CF^+, comCF^+ \) are incomparable).

a) \( LN(0) = CF^+ \).

The rules of a FODG not allowing any gaps are interpreted in an usual context-free way.

b) \( dLN(0) \) contains non-context-free languages.

Such a language is, e.g., \( L(G_2) \) from example 2. It holds that all De-trees from \( dT(G_2) \) do not have any gaps. On the other hand the number of gaps in the set of DR-trees parsed by \( G_2 \) is not bounded by any constant.

c) \( LN(\ast) \) is not a context-free language.

It easy to see the inclusion. The fact that it is a proper inclusion can be shown by the context-free language \{\( a^n b^m c^m a^n \mid n, m > 0 \}\), a language which is not commutative context-free. and is obviously from \( LN(i) \) for any \( i \in (\text{Nat} \cup \{0\}) \cup \{\ast\} \).

d) The \( L(G_1), L(G_2) \) from the example 2 are commutative CF-languages which are not context-free.

The language \( L_C = \{a^n b^m c^m a^n \mid n, m > 0 \} \) is from \( CF^+ \), but it is not from \( LN(\ast) \).

It means that the classes \( LN(\ast) \) and \( LN(0) = CF^+ \) are incomparable.

e) \( LN(i) \subseteq dLN(i) \) for any natural number \( i \)

since \( \text{Ng}(Tr) \geq d\text{Ng}(dT(Tr)) \) for any DR-tree \( Tr \).

f) Any language \( L \) from \( LN(i) \), where \( i \in (\text{Nat} \cup \{0\}) \), is recognizable in a polynomial time compared to the size of the input.

We have implemented (see [Holan.Kuboa, Pla'tek, 1995], [Holan.Kuboa, Pla'tek, 1997]) a natural bottom-up parsing algorithm based on the stepwise computation of pairs of the shape \( (U, Cr) \), where \( U \) means a node of a DR-tree and \( Cr \) means its coverage. With the \( \text{Ng}(i) \) it can be interpreted as the maximum of the number of gaps in the coverages during a computation of the parser limited by a constant. The number
of such pairs depends polynomially on the size of the input. The assertion f) is derived from this observation.

g) Because a limited \( dNg \) for a language \( L \) does not ensure also the limited \( Ng \) (see the item b)) for \( L \), the limited \( dNg \) need not ensure the parsing in a polynomial time for the language \( L \).

5 A measure of word order freedom of syntactically correct Czech sentences

In this section we describe linguistic observations concerning the word-order complexity of the surface Czech syntax. The notions defined above are used in this section. We are going to discuss the fact that for Czech syntax (described by means of a FODG \( GE \)) there is no adequate upper estimate of the bound of word-order freedom based on node-gaps complexities. It means that we are going to show that there is no \( i_0 \) such that each De-tree belonging to \( dTN(GE,i) - dTN(GE,i_0) \) for \( i > i_0 \) is (quite clearly) syntactically incorrect.

Let us now show a number of examples of non-projective constructions in Czech. In the previous work (see [Holan,Kubůn,Plátek,1997]) we have put forward a hypothesis that from the practical point of view it is advisable to restrict the (possible) local number of gaps to one. However, we found out soon that this is not generally true because it is not very difficult to find a perfectly natural, un-understandable and syntactically well-formed sentence with \( dNg \) higher than one. Such a sentence may for example look like this:

\[ Tuto knihu jsem se mu rozhodl dát k narozeninám. \]

(Lit.: This book l-have-Refl. him decided to give to birthday.)

[I decided to give him this book to birthday.]

The Fig.3 shows one of possible De-trees representing this sentence:

The node[\( dšt, 7,6 \)] has a coverage containing two gaps. Since no other node has a coverage with more gaps, then (according to the definition of \( dNg \)) the dependency node-gaps complexity of this dependency tree is equal to 2. It is even quite clear that it is not possible to find a linguistically adequate De-tree representing the same sentence with a lower value of \( dNg \) (words \( jsem, se \) and \( rozhodl \) will always cause gaps in the coverage of \( dšt \)). The maximum empirically attested number of verbal participants is 5 in Czech (see [Sgall,Panovevá,1988/89]).

The previous example showed that the infinitives of verbs with such a number of participants may quite naturally form constructions containing four gaps. It might even suggest that the number four might, hence, serve as an upper estimation of \( dNg \) (based on the highest possible number of participants of an infinitive). However, this conclusion would not be correct, since in the general case it is necessary to take into account also the possibility that participants are combined with free modifiers, and from this it follows that it is not reasonable to set a certain constant as an upper boundary of \( dNg \). In order to exemplify this, we present the sentence

\[ Za dnešní krize by se lidem jako Petr kvůli jejich příjmu takový byt žádný majitel domu za tu cenu nemohl na tak dlouhou dobu snažit pronajímat. \]

(Lit.: In today's crisis would Refl. people (dat.) as Petr because of their income such flat no owner of-a-house for that price couldn't for such long time try to-rent)

[In today's crisis no landlord would try to rent a flat for that price for such a long time to such people as Petr because of their income.]

Similarly as in the previous case. Fig.5 shows a dependency tree representing the structure of the sample sentence. The words by, příjmu, majitel, snažit and pronajímat depend (immediately) on the main verb nemohl. They are together with their dependents interlocked with the set of word groups {\( Za dnešní krize; lidem jako Petr: takový byt: za tu cenu: na tak dlouhou dobu \) dependent on the subordinated verb in infinitive form (pronajímat - to rent), creating thus five gaps in the local tree whose governor is pronajímat.

The previous claim is supported also by another example of combination of different types of non-projectivities inside one clause. Apart from gaps caused by complementations, the following syntactically correct sentence contains also a gap between a wh-pronoun and a noun. The Fig.4 shows that the value of \( dNg \) is equal to 3 for this sentence.

\[ Ke kolikýmým jsem se mu nakonec tuto knihu rozhodl dát narozeninám? \]

(Lit.: To which I-have-Refl. him finally this book decided to give birthday)

[Which birthday I finally decided to give him this book to?]

The examples presented above illustrate the fact that the measure of \( dNg \) has (in Czech) similar properties as some other simple measures of complexity of sentences (the length of a sentence, the level of center-embedding, etc.). It is quite clear that for the sake of readability and simplicity it is advisable to produce sentences with a low score of these simple measures. On the other hand, it is not possible to
Figure 3:
Tuto knihu jsem se mu rozhodli dát k narozeninám.
This book I have him decided to give to birthday.

Figure 4:
Ke kolikým jsem se mu nakonec tuto knihu rozhodli dát narozeninám?
To which I have him finally this book decided to give birthday?
set a fixed threshold above which the sentence may be considered ungrammatical. Similarly as very long sentences or sentences with complex self-embedding, also the sentences containing a large number of gaps may be considered stylistically inappropriate, but not syntactically ill-formed.

6 An enhancing of FODG

Even when the examples presented above show the unlimited nature of nonprojective constructions in Czech, it is necessary to point out that there are some constructions, which do not allow nonprojectivity at all. These constructions therefore require a limitation applied to particular grammar rules handling them. For example, some of the grammar rules then allow only projective interpretation of input data. As an example of such a construction in Czech we may take an attachment of a nominal modifier in genitive case to a governing noun (obchod[nom.] oce[gen.] - shop-of-father, father's shop). Only the words modifying (directly or indirectly) one of the nouns may appear in between the two nouns. no "gap" is allowed.

(obchod s potravinami našeho oce)

Formally we add the previously described "projectivity" constraint to a rule of the form $A \rightarrow x \ CB$ with help of an upper index $0$. Hence the augmented rule has the following form $A \rightarrow x^0 \ CB$. It means that any node of a DR-tree created by this rule (corresponding to the left-hand side $A$) can be a root of a subtree without gaps only.

We can easily see that with such a type of rules, the power of (enhanced) FODG's increases substantially.

Formal observation. The class of languages parsed (generated) by enhanced FODG's contains $CF^+$ and $comCF^+$. This type of enhanced FODG is used for our current robust parser of Czech.

We outline a further enhancing of FODG's: We can imagine that if we write for some natural number $i$ a rule of the shape $A \rightarrow x^i \ CB$ it means that any node created by this rule can have a coverage with at most $i$ gaps. We can easily see that with such a type of rules, the power of FODG's enhanced in this way again increases substantially.

7 Conclusion

We have not considered some other natural measures of non-projectivity here, e.g., the number of crossings ([Kunze.1972]), the size of gaps or the tree-gaps complexity ([Holans, Kuboň, Plátek.1985]). By this measure it is much easier to show the non-existence of their upper boundaries concerning the non-projectivity in Czech than by $dXg$. The presented measures $Xg$ and $dXg$ help us to show that it is not easy to describe the Czech syntax (in particular, to reflect the word-order) fully and adequately. It is further shown that it is not easy (maybe impossible) to find an algorithm which should guarantee parsing of any correct Czech sentence with a (not too high) polynomial time complexity according to its size. We will try in future to improve the description, and also the parsing algorithm, in order to come closer to meeting this challenge.

8 Acknowledgement

We are very thankful to prof. Hajčová and prof. Panevová for their recommendations.

The work on this paper is a part of the project "Formal Tools for Separation of Syntaxically Correct Structures from Syntaxically Incorrect Structures" supported by the Grant Agency of the Czech Republic under Grant No. 201/96/0195.

References

M.I.Beleckij: Beskontekstnye i dominacionnye grammatiki i svyazannye s nimi algoritmičeskije problemy, Kibernetika, 1967, No 4, pp. 90-97
H. Gaifman: Dependency Systems and Phrase-Structure Systems, Information and Control, 8, 1965, pp. 301-337
A.V. Gladkij: Formal'nye grammatiki i jazyki, iz.: NAUKA, Moskva, 1973
D.T.Hugnh: Commutative Grammars: The complexity of uniform word problems. Information and Control 57, 1983, pp. 21-39
T.Holan, V.Kuboň, M.Plátek: A Grammar-Checker for Czech. Proceedings of the Fifth Conference on Applied Natural Language Processing, ed. Association for Computational Linguistics, Washington, March 1997, pp. 147-154
V.Kuboň, T.Holan, M.Plátek: A Grammar-Checker for Czech. UFA L Tech. Report TR-1997-02, MFF Charles University Prague, 1997
J.Kunze: Die Auslassbarkeit von Satzteilen bei koordinativen Verbindungen in Deutschen. Akademie-Verlag-Berlin, 1972
I.A.McCuk: Dependency Syntax: Theory and Practice. State University of New York Press, 1988
M.Plátek: Questions of Graphs and Automata in Generative Description of Language. The Prague Bulletin of Mathematical Linguistics 21,1974, pp. 27-63
P.Sgall, J.Pancirová: Dependency Syntax - a Challenge. Theoretical Linguistics, Vol. 15, No. 1/2, Walter de Gruyter, Berlin - New York 1988/89, pp. 71 - 86

28