Characterizing Discontinuity in Constituent Treebanks

Wolfgang Maier and Timm Lichte

University of Tübingen
{wo.maier,timm.lichte}@uni-tuebingen.de

Abstract. Measures for the degree of non-projectivity of dependency grammar have received attention both on the formal and on the empirical side. The empirical characterization of discontinuity in constituent treebanks annotated with crossing branches has nevertheless been neglected so far. In this paper, we present two measures for the characterization of both the discontinuity of constituent structures and the non-projectivity of dependency structures. An empirical evaluation on German data as well as an investigation of the relation between our measures and grammars extracted from treebanks shows their relevance.

1 Introduction

Discontinuous phrases are common in natural language. They occur particularly frequently in languages with a relatively free word order like German, but also in fixed word order languages, like English or Chinese. Treebank annotation must account for sentences containing such phrases. In constituent treebanks, however, a direct annotation of discontinuities which would require crossing branches is generally not allowed. Instead, an annotation backbone based on context-free grammar is extended by a labeling mechanism, sometimes in combination with traces, which accounts for discontinuous phrases. Examples for such treebanks include the Penn Treebank (PTB) [1], the Spanish 3LB [2], and the German TüBa-D/Z [3].

The German NeGra [4] and TIGER [5] treebanks are among the few notable exceptions to this kind of annotation, where crossing branches are allowed and discontinuous constituents are annotated directly. As an illustration, Fig. 1 shows the constituent annotation of (1), a sentence from NeGra involving two discontinuous VPs. The discontinuity is due to the leftward extraposition of the PP object *Darüber*.

(1) Darüber muß nachgedacht werden.
Thereof must thought be.
"Thereof must be thought."

Measures for the degree of discontinuity of constituent trees with crossing branches in these treebanks have not been subject to research, but would warrant an investigation. First, linguistic phenomena like long-distance dependencies, which give rise to discontinuous structures, are often assumed not being

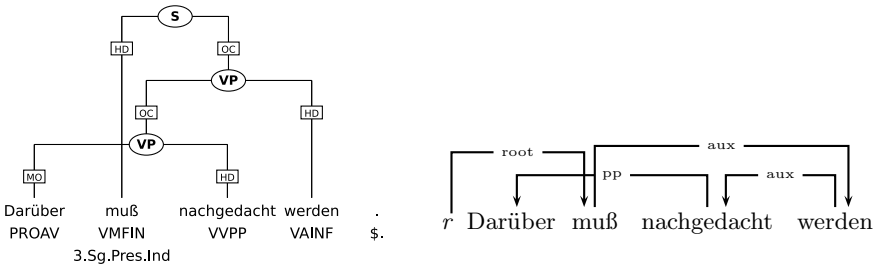

Fig. 1. Constituent and dependency analysis for a NeGra sentence

totally unrestricted as are the crossing edges in treebank annotation. Furthermore, in application contexts like statistical parsing, where treebank trees are interpreted as derivation structures of a grammar formalism, the information encoded by the crossing branches is simply discarded in order to avoid processing difficulties: Commonly the trees are transformed into non-crossing structures and a (probabilistic) context-free grammar is read off the resulting trees [6, 7].

The situation for dependency grammar is different. Dependency treebanks generally contain *non-projective* graphs which, in contrast to *projective* graphs, contain nodes with an discontinuous yield (see Fig. 1 for a dependency analysis of (1)). In recent years, motivated by processing concerns, particularly the formal characterization of non-projectivity has received attention, see [8] for a survey. Among other measures, *gap degree* [9] and *well-nestedness* [10] have emerged. Both of them were shown to empirically describe non-projective dependency data well. They can furthermore be exploited for parsing [11].

The contribution of this paper is to give a characterization of gap degree and well-nestedness of constituent trees with the same descriptive and practical relevance as the corresponding concepts in dependency grammar. For this purpose, we show that both gap degree and well-nestedness can be formulated as equivalent properties of different kinds of graphs, more precisely, of dependency graphs and constituent trees. We explore the relation between our measures and grammars extracted from dependency and constituent treebanks and conduct an empirical investigation on German data shows the practical relevance of our measures.

In the following section, we introduce our new definitions of gap degree and well-nestedness. In Sect. 3 we investigate relation between our measure and grammars extracted from treebanks. In Sect. 4, we empirically review the measures for non-projectivity of dependency graphs and discontinuity of constituent trees on German data. Sect. 5 summarizes the article and presents future work.

2 Discontinuity Measures

In the following, we define both dependency structures and constituent structures in terms of trees, followed by the definitions of gap degree and well-nestedness.

2.1 Syntactic Structures

Definition 1 (Tree). *Let $D = (V, E, r)$ be a directed graph with V a set of nodes, $E : V \times V$ a set of arcs and $r \in V$ the root node. D is a* labeled tree *iff*

1. *D acyclic and connected;*
2. *All nodes $V \setminus \{r\}$ have in-degree 1, r has in-degree 0;*
3. *D disposes of two disjoint alphabets L_V, L_E of node and edge labels and the corresponding labeling functions $l_V : V \to L_V$ and $l_E : E \to L_E$.*

The nodes with out-degree 0 are called leaves. *We write $v_1 \to v_2$ for $\langle v_1, v_2 \rangle \in E$. $\overset{*}{\to}$ is the reflexive transitive closure of E. We say that v_1* dominates *v_2 iff $v_1 \overset{*}{\to} v_2$.*

On the basis of trees, we define *dependency structures* and *constituent structures.*

Definition 2 (Dependency structure[1]). *A* dependency structure *for a sentence $s = w_1 \cdots w_n$ is a labeled tree $D_{dep} = (V, E, r)$ with the labeling functions l_V and l_E, such that*

- *$l_V : V \to \{0, \ldots, n\}$, $l_V(v) \neq 0$ for all $v \in V \setminus \{r\}$, $l_V(r) = 0$,*
- *$l_E : E \to L_E$, L_E being a set of dependency edge labels.*

Note that the root node in dependency structures is mapped to 0, which is no terminal position and is therefore not included in its yield (see below).

Definition 3 (Constituent structure). *A* constituent structure *for a sentence $s = w_1 \cdots w_n$ is a labeled tree $D_{con} = (V, E, r)$ with the labeling functions l_V and l_E, such that*

- *$l_V : V \to \{1, \ldots, n\} \cup N$, N being a set of syntactic category labels disjoint from $\{1, \ldots, n\}$, such that for a $v \in V$, $l(v) \in \{1, \ldots n\}$ if v is a leaf and $l(v) \in N$ otherwise,*
- *$l_E : E \to L_E$, L_E being a set of grammatical function labels.*

Definition 4 (Syntactic structure). *We call $D = (V, E, r)$ a* syntactic structure *if it is either a dependency structure or a constituent structure. The* yield *π_v of a $v \in V$ is the set $\{i \in \mathbb{N}^+ \mid$ there is a $v' \in V$ such that $v \overset{*}{\to} v'$ and $l(v') = i\}$.*

2.2 Discontinuity Measures

Both gap degree and well-nestedness can be defined on the yields of syntactic structures. Intuitively, gap degree is a measure for the amount of discontinuity in syntactic structures and a gap is a discontinuity in the yield of a node.

Definition 5 (Gap degree). *Let $D_{syn} = (D, l)$ be a syntactic structure with $D = (V, E, r)$.*

[1] Our definition follows the definition of *Abhängigkeitsbäumen* (dependency trees) in [12].

1. *Let π_v be the yield of a $v \in V$. A* gap *is a pair (i_n, i_m) with $i_n, i_m \in \pi_v$ and $i_n + 2 \leq i_m$ such that there is no $i_k \in \pi_v$ with $i_n < i_k < i_m$. The* gap degree *d of a node v in a syntactic structure is the number of gaps in π_v,*
2. *The gap degree d of a syntactic structure D_{syn} is the maximal gap degree of any of its nodes.*

A dependency structure with gap degree > 0 is called non-projective*; a constituent structure with gap degree > 0 is called* discontinuous*.*

In other words, the gap degree corresponds to the maximal number of times the yield of a node is interrupted. n gaps of a node entail $n + 1$ uninterrupted yield intervals. This is reflected in the measure of *block degree* [8, pp. 35], which is in fact the gap degree plus 1. Fig. 2 shows an example. Both syntactic structures in D_{dep-g} and D_{con-g} have gap degree 1. In D_{dep-g} both yields π_{v_4} and π_{v_5} have the maximal gap degree 1 due to the fact that they do not contain 3. In D_{con-g}, v_1 has gap degree 1 because 2 is not included in its yield. D_{dep} and D_{con} in Fig. 3 both have gap degree 0.

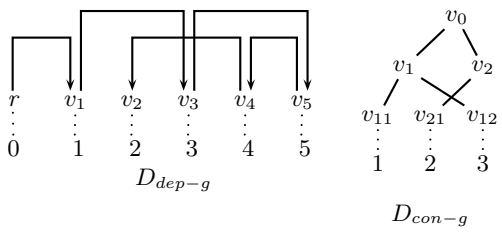

Fig. 2. Gap degree in syntactic structures

We now define *well-nestedness* (as opposed to *ill-nestedness*) as another property of syntactic structures. Intuitively, in a well-nested structure, it holds for all nodes which do not stand in a dominance relation that their yields do not interleave.

Definition 6 (Well-Nestedness[2]). *Let $D_{syn} = (D, l)$ be a syntactic structure. D_{syn} is* well-nested *iff there are no disjoint yields π_{v_1}, π_{v_2} of nodes $v_1, v_2 \in V$ such that for some $i_1, i_2 \in \pi_{v_1}$ and $j_1, j_2 \in \pi_{v_2}$ it holds that $i_1 < j_1 < i_2 < j_2$.*

It is easy to see that an ill-nested syntactic structure must necessarily have a gap degree ≥ 1. Fig. 3 shows well-nested and ill-nested constituent structures and dependency structures. D_{con} and D_{dep} are well-nested, while D_{con-n} and D_{dep-n} are not. Note that, while D_{con} and D_{dep} have gap degree 0 in addition to being well-nested, both D_{con-g} and D_{dep-g} in Fig. 2 are well-nested but have a gap degree greater than 1.

[2] If the considered yields are not restricted to be disjoint, the syntactic structures are called *planar* or *weak non-projective* (see [13, 14]).

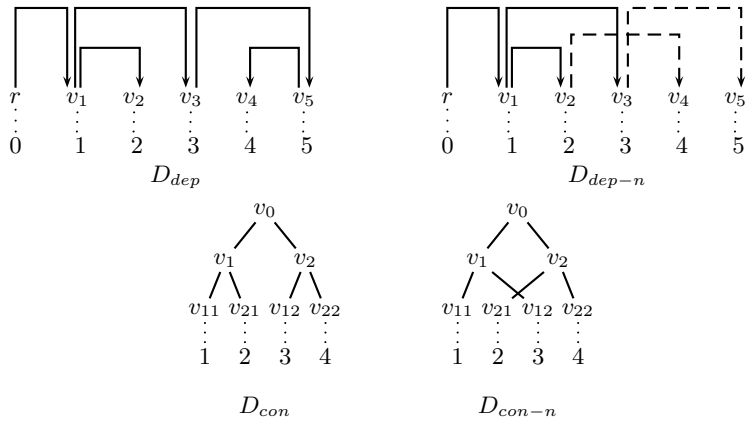

Fig. 3. Well-nestedness and ill-nestedness in syntactic structures

Gap degree and well-nestedness had not been defined for constituent trees before. However, with respect to dependency structures our definitions do correspond to the definitions of gap degree and well-nestedness from the literature [9, 10, 14].

The definition of well-nestedness (resp. ill-nestedness) does not provide a notion of a degree. For this purpose, to our knowledge, two measure have been introduced so far in the context of dependency structures. [13] introduces *level types* and [15] introduce *strongly ill-nested structures*. While both measures do characterize well the data considered in the resp. articles, the first measure is exclusively motivated by dependency grammar and the second one only discriminates very complex dependency structures which are unlikely to occur in a natural language context due to their sheer complexity. In the following, we define a new measure called *k-ill-nestedness* which we intend to intuitively capture the degree of interleaving of yields in constituent structures and dependency structures. k stands for the number of disjoint yields that interleave with some other single yield which is disjoint from them.

Definition 7 (k-Ill-Nestedness). *Let $D_{syn} = (D, l)$ be an ill-nested syntactic structure. D is said to be* k-ill-nested *iff there exist disjoint yields $\pi_v, \pi_{v_1}, \ldots, \pi_{v_k}$ of nodes $v, v_1, \ldots, v_k$ in D such that for some $i_1, i_2 \in \pi_v$ and $j_1^{(m)}, j_2^{(m)} \in \pi_m$, $1 \leq m \leq k$, it holds that $i_1 < j_1^{(m)} < i_2 < j_2^{(m)}$ or $j_1^{(m)} < i_1 < j_2^{(m)} < i_2$.*

Note that well-nestedness as defined above is equivalent to 0-ill-nestedness. The dependency structure D_{dep-n} and the constituent structure D_{con-n} in Fig. 3 are 1-ill-nested. An example for 2-ill-nested syntactic structures is D_{con-lr} in Fig. 4.

2.3 Further properties of constituent structures

We now give a further formal characterization of constituent structures. We first define the *maximal nodes* of a gap. Informally speaking, the entire yield of a maximal node lies in the gap, but the yield of its parent node does not.

Definition 8 (Maximal node). *Let $D_{con} = (V, E, r)$ be a constituent structure, π_v the yield of a $v \in V$ and (i_1, i_2) a gap in π_v. Then $v_{max} \in V$ is a* maximal node *of (i_1, i_2) iff*

1. *for all $j \in \pi_{v_{max}}$, it holds that $i_1 < j < i_2$,*
2. *there is a node $u \in V$ with the yield π_u such that $u \rightarrow v_{max}$, and there is a $k \in \pi_u$ with $k \leq i_1$ or $k \geq i_2$.*

The node u is called gap filler*.*

A gap (i_1, i_2) can have more than one maximal node, and the combined yield of all maximal nodes is the set $\{i \mid i_1 < i < i_2\}$. As an example, consider D_{con-n} in Fig. 3. The only maximal node of the gap $(1, 3)$ of v_1 is v_{21}.

Intuitively, in a well-nested constituent structure $D_{con} = (V, E, r)$, all gaps are filled from "above". That means that all gap fillers (i.e., the parent nodes of all maximal nodes) of all gaps (i_1, i_2) in the yield of a $v \in V$ immediately dominate v itself. As an example, compare the well-nested structure D_{con-g} (Fig. 2) with the ill-nested structure D_{con-n} (Fig. 3): In the first, v_0 is the maximal node of the gap of v_1 and $v_0 \rightarrow v_1$, while in D_{con-n}, v_2 is the maximal node of the gap of v_1 and $v_2 \stackrel{+}{\not\rightarrow} v_1$. We formalize this intuition in Lemma 1.

Lemma 1. *Let $D_{con} = (V, E, r)$ be a well-nested constituent structure, $v \in V$ a node with gap degree ≥ 1, (i_1, i_2) a gap in π_v and v_{max} a maximal node of (i_1, i_2). There is no node v' with $v' \rightarrow v_{max}$ and $v' \stackrel{+}{\not\rightarrow} v$.*

Proof. By contradiction. Assume that there is a node v' with $v' \rightarrow v_{max}$ and $v' \stackrel{+}{\not\rightarrow} v$. According to the definition of maximal nodes, there must be a $k_i \in \pi_{v'}$ with $k_i < i_1$ or $k_i > i_2$ and a k_j with $i_1 < k_j < i_2$. That means that it is either the case that $k_i < i_1 < k_j < i_2$ or $i_1 < k_j < i_2 < k_i$, both of which contradict the definition of well-nestedness. □

Ill-nested constituent structures are thus constituent structures in which some gap is filled from a direction other than "above", in other words, in which the gap filler does not dominate the node with the gap. Gaps in such structures can intuitively be filled from the left, from the right, or from both sides. If we assume an implicit ordering of the nodes based on the smallest number in their yield, this intuition holds even though we deal with unordered trees. Such an ordering is in fact the basis for all visual representations of constituent structures in this paper. Note that if a gap is filled from the left or the right and additionally from above, we are still dealing with an ill-nested structure (cf. Lemma 1).

Definition 9 (Gap filler locations). *Let $D_{con} = (V, E, r)$ be an ill-nested constituent structure, $v \in V$ a node with gap degree ≥ 1, (i_1, i_2) a gap in π_v and v_{max} a maximal node of (i_1, i_2). We say that*

1. *(i_1, i_2) is* filled from the left *iff there is a node v' with the yield $\pi_{v'}$ and $v' \rightarrow v_{max}$ such that there is an $i_l \in \pi_{v'}$ with $i_l < i_1$ and*
2. *(i_1, i_2) is* filled from the right *iff there is a node v' with the yield $\pi_{v'}$ and $v' \rightarrow v_{max}$ such that there is an $i_l \in \pi_{v'}$ with $i_l > i_1$.*

Fig. 4 shows three example trees with different gap filler locations.

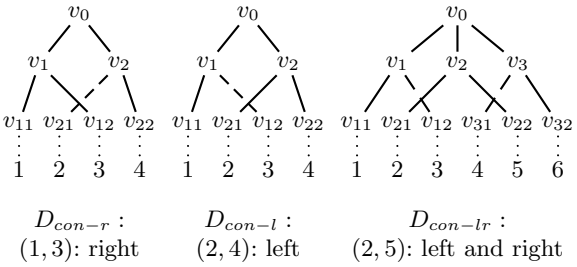

Fig. 4. Gap filler position in ill-nested constituent structures

3 Properties of Extracted Grammars

Both non-projective dependency structures and discontinuous constituent structures, as occurring in treebanks, can be interpreted as derivation structures of linear context-free rewriting systems (LCFRS) [16], resp. of the equivalent formalism of simple range concatenation grammar (sRCG) [17]. [18] present a grammar extraction algorithm for constituent structures and [11] presents an extraction algorithm for dependency structures. In the following we resume previous work by presenting simple RCG together with an extraction algorithm for this formalism on syntactic structures.

3.1 Simple Range Concatenation Grammar

Definition 10 (Ordered simple k-RCG [17]). *A* simple RCG *is a tuple* $G = (N, T, V, P, S)$ *where* N *is a finite set of predicate names with an arity function dim:* $N \to \mathbb{N}$, T *and* V *are disjoint finite sets of terminals and variables,* P *is a finite set of clauses of the form*

$$A(\alpha_1, \ldots, \alpha_{dim(A)}) \to A_1(X_1^{(1)}, \ldots, X_{dim(A_1)}^{(1)}) \cdots A_m(X_1^{(m)}, \ldots, X_{dim(A_m)}^{(m)})$$

for $m \geq 0$ *where* $A, A_1, \ldots, A_m \in N$, $X_j^{(i)} \in V$ *for* $1 \leq i \leq m, 1 \leq j \leq dim(A_i)$ *and* $\alpha_i \in (T \cup V)^*$ *for* $1 \leq i \leq dim(A)$, *and* $S \in N$ *is the start predicate name with* $dim(S) = 1$. *For all* $p \in P$, *it holds that every variable* X *occurring in* c *occurs exactly once in the left-hand side (LHS) and exactly once in the RHS. A simple RCG is* ordered *if for all* $p \in P$, *it holds that if a variable* X_1 *precedes a variable* X_2 *in some RHS predicate, then* X_1 *also precedes* X_2 *in the LHS predicate. A simple RCG* $G = (N, T, V, P, S)$ *is a* k-RCG *if for all* $A \in N$, $dim(A) \leq k$. k *is called the* fan-out *of* G. *The* rank *of* G *is the maximal number of predicates on the RHS of one of its clauses.*

For the definition of the language of a simple RCG, we borrow the LCFRS definitions.

Definition 11 (Language of a simple RCG). *Let* $G = \langle N, T, V, P, S \rangle$ *be a simple RCG. For every* $A \in N$, *we define the yield of* A, $yield(A)$ *as follows:*

1. *For every* $A(\boldsymbol{\alpha}) \to \varepsilon$, $\boldsymbol{\alpha} \in yield(A)$*;*
2. *For every clause*

$$A(\alpha_1, \ldots, \alpha_{dim(A)}) \to A_1(X_1^{(1)}, \ldots, X_{dim(A_1)}^{(1)}) \cdots A_m(X_1^{(m)}, \ldots, X_{dim(A_m)}^{(m)})$$

and all $\boldsymbol{\tau_i} \in yield(A_i)$ *for* $1 \leq i \leq m$, $\langle f(\alpha_1), \ldots, f(\alpha_{dim(A)})\rangle \in yield(A)$ *where* f *is defined as follows:*
(a) $f(t) = t$ *for all* $t \in T$,
(b) $f(X_j^{(i)}) = \boldsymbol{\tau_i}(j)$ *for all* $1 \leq i \leq m, 1 \leq j \leq dim(A_i)$ *and*
(c) $f(xy) = f(x)f(y)$ *for all* $x, y \in (T \cup V)^+$.
3. *Nothing else is in* $yield(A)$.

The language is then $\{w \mid \langle w\rangle \in yield(S)\}$.

3.2 Grammar Extraction

We can resume both of the afore-cited grammar extraction algorithms in a single algorithm. We extract a simple RCG G from a syntactic structure $D = (V, E, r)$ over a sentence $w_1 \ldots w_n$ as follows. For each $v_0 \in V$ with the children $v_1, \ldots, v_k$, $k \geq 0$, we construct a clause $p = \psi_0 \to \psi_1 \cdots \psi_k$, where ψ_i, $0 \leq i \leq k$, are predicates which we construct as follows. If D is a dependency structure, then ψ_0 receives the label of the incoming arc of v and ψ_i, $1 \leq i \leq k$, receives the label of the arc between v and v_i; otherwise, ψ_i, $0 \leq i \leq k$, receives the node label of v_i. To the name of each ψ_i, $0 \leq i \leq k$, we append the block degree of v_i. In order to determine the predicate arguments, for each $u \in \pi_0$, we introduce a variable X_u. Then for all $\psi_i, 0 \leq i \leq k$, the following conditions must hold.

1. The concatenation of all arguments of ψ_i is the concatenation of all $\{X_u \mid u \in \pi_i\}$ such that for all $p, q \in \pi_i, p < q$, it holds that X_p precedes X_q.
2. A variable X_i with $1 \leq i < n$ is the right boundary of an argument of ψ_i iff $i + 1 \notin \pi_i$, i.e., and argument boundary is introduced at each gap.

If D is a dependency structure and $l(v_0) = i$, we exchange the variable X_i in ψ_0 with w_i. If D is a constituent structure, $l(v_0) = i$ and $l(v') = t_i$, $v' \to v$, we add a add a so-called *lexical clause* $t_i(w_i) \to \varepsilon$ to the grammar, t_i being the part-of-speech tag of w_i. Sequences of more than one variable which are a RHS argument are collapsed into a single variable both on the LHS and the RHS of the clause. The completed clause p is added to the grammar.

As an example, Fig. 5 shows the two grammars extracted from the constituent and dependency structures in Fig. 1.

3.3 Grammar Properties

The gap degree and the ill-nestedness of the syntactic structure used for grammar extraction determine certain properties of the extracted grammar. The properties in Corollary 1 are immediately obvious.

S1($X_1X_2X_3$) → VP2(X_1,X_3) VMFIN1(X_2)
VP2(X_1, X_2X_3) → VP2(X_1,X_2) VAINF1(X_3)
VP2(X_1, X_2) → PROAV1(X_1) VVPP1(X_2)
PROAV(Darüber) → ε, VMFIN(muß) → ε
VVPP(nachgedacht) → ε, VAINF(werden) → ε.

pp(Darüber) → ε
root(X_1mußX_3) → aux(X_1,X_3)
aux(X_1,nachgedacht) → pp(X_1)
aux(X_1, X_2werden) → aux(X_1,X_2)

Fig. 5. Simple RCGs obtained from constituent structures (left) and from dependency structures (right)

Corollary 1 (Grammar Properties). *Let G be a simple RCG of rank r and fan-out g. A syntactic structure D derived from G has gap degree $g-1$ and does not have nodes with more than r children.*

Assuming the relation between simple RCGs and syntactic structures given by the grammar extraction algorithm above, we can also draw conclusions from the clauses of a simple RCG G on the well-nestedness of its derivations. Lemma 2 relates the interleaving of the variables in the arguments of clauses with the conditions given by the definition of well-nestedness. For simplicity, we assume that all clauses $p \in P$ contain only continuously numbered variables X_1 through X_m in their LHS predicate. Furthermore, we introduce the function $\eta : P \times \mathbb{N} \rightarrow \mathbb{N}^+$, which returns the numbers of the variables used in the arguments of the ith RHS predicate of a clause.

Lemma 2 (Ill-Nestedness and Grammars). *Let G be a ordered simple RCG which produces k-ill-nested derivations with $k \geq 1$. There is a clause $p \in P$, $p = \psi_0 \rightarrow \psi_1 \cdots \psi_m$ such that there is a pair of predicates (ψ_i, ψ_j), $1 \leq i < j \leq m$ for which it holds that $i_1, i_2 \in \eta(p, i)$ and $j_1, j_2 \in \eta(p, j)$ with $i_1 < j_1 < i_2 < j_2$.*

Proof. By contradiction. Assume there is no such clause and G produces ill-nested derivations. Due to the definition of well-nestedness, there must be a derivation $D = (V, E, r)$ of G with disjoint yields π_{v_1}, π_{v_2} of nodes $v_1, v_2 \in V$ such that for some $i_1, i_2 \in \pi_{v_1}$ and $j_1, j_2 \in \pi_{v_2}$ it holds that $i_1 < j_1 < i_2 < j_2$. The fact that π_{v_1} and π_{v_2} are disjoint entails that v_1 and v_2 do not dominate each other. Furthermore, it entails that the yield π_{lca} of their least common ancestor v_{lca} must contain both π_{v_1} and π_{v_2} and that for all yields π' of nodes v' with $v_{lca} \xrightarrow{*} v'$, it holds that $\pi' \cap \pi_{v_1} = \emptyset$ or $\pi' \cap \pi_{v_2} = \emptyset$. Assume v_{lca} and its children have been generated by a clause $\psi_{lca} \rightarrow \psi^1_{lca} \cdots \psi^m_{lca}$. Due to the aforementioned condition on the yields of all nodes dominated by v_{lca} and condition 1 in the extraction algorithm, there must be at least two predicates $\psi^i_{lca}, \psi^j_{lca}$, $1 \leq i < j \leq m$ for which it holds that $i_1, i_2 \in \eta(p, i)$ and $j_1, j_2 \in \eta(p, j)$ with $i_1 < j_1 < i_2 < j_2$. This contradicts our initial assumption. □

4 Empirical Investigation

4.1 Non-Projective Dependency Structures

The empirical relevance of gap degree and well-nestedness has been shown for dependency treebanks in works such as [14] on the basis of the Prague Depen-

dency Treebank (PDT) [19] and the Danish Dependency Treebank (DDT) [20]. We have extended these previous investigations on two other treebanks, namely the dependency versions of the NeGra and TIGER, which we will call TIGER-Dep and NeGra-Dep. Both have been built with the dependency converter for TIGER-style trees from [21]. We are aware of the fact that all dependency conversion methods introduce undesired noise, but we choose this well-established method rather than using other German dependency data sets like the CoNLL-X TIGER data (used by [13]) or the very small TIGER-DP [22] due to our desire of obtaining comparable dependency structures for both constituent treebanks. Since punctuation is generally not attached to the trees and therefore not part of the annotation, it has been removed in a preprocessing step from all trees in both treebanks prior to the conversion. This lead to the exclusion of a handful of sentences from TIGER and NeGra, since they consisted only of punctuation. Tab. 1 contains the gap degree figures and ratios of well-nestedness of the PDT and the DDT, borrowed from [14], and our findings for NeGra-Dep and TIGER-Dep.

	DDT		PDT		NeGra-Dep		TIGER-Dep	
number of sent.	4393		73088		20597		40013	
av. sent. length	18		17		15		16	
gap degree 0	3732	84.95%	56168	76.85%	16695	81.06%	32079	80.17%
gap degree 1	654	14.89%	16608	22.72%	3662	17.78%	7466	18.66%
gap degree 2	7	0.16%	307	0.42%	225	1.09%	438	1.09%
gap degree 3	–	–	4	0.01%	12	0.05%	22	0.05%
gap degree 4	–	–	1	< 0.01%	2	0.01%	4	0.01%
gap degree ≥5	–	–	–	–	1	< 0.01%	4	0.01%
well-nested	4388	99.89%	73010	99.89%	20472	99.39%	39750	99.34%
1-ill-nested	?		?		125	0.61%	263	0.66%

Table 1. Gap degree and well-nestedness in DDT and PDT (figures from [14]) and in NeGra-Dep and TIGER-Dep

The gap degree figures of NeGra-Dep and TIGER-Dep lie in the same range as the figures of DDT and PDT. A closer look at the dependency annotations of TIGER-Dep and NeGra-Dep reveals that the most common causes for a high gap-degree (≥ 3) are enumerations, appositions and parenthetical constructions.

As [8, p. 62] remarks, the well-nestedness constraint seems to be a good extension of projectivity since almost all dependency structures found in treebanks adhere to it. This is confirmed by our quantitative findings in the German treebanks. The structures which do not adhere to the well-nestedness constraint are all 1-ill-nested. Note that 1-ill-nestedness is a stronger constraint than non-well-nestedness. A closer linguistic inspection of the ill-nested dependency structures in Negra-Dep and TIGER-Dep shows that most common reason for ill-nestedness is not erroneous annotation etc., but linguistically acceptable analyses of extraposition phenomena. Fig. 6 shows a typical ill-nested dependency analysis of a

sentence of NeGra. The edges relevant for the ill-nestedness are dashed. In this sentence, the subject noun *ein Modus* dominates a non-adjacent, extraposed relative clause, while being surrounded by a disjoint subtree, namely the non-finite main verb *eingespielt* and its dependent. The annotation can be related

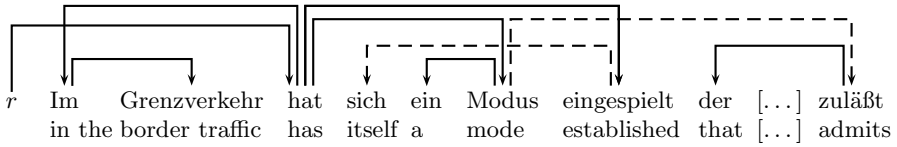

Fig. 6. An ill-nested dependency structure in Negra-Dep

to linguistic generalizations on dependency structures discussed in the literature (particularly on German):

1. The subject depends on the finite verb [12, p. 110][23, pp. 83].
2. The non-finite verb depends on the finite verb and governs its objects and modifying expressions[24, p. 189]. In our treebanks, this is only true for objects and modifying expressions outside the *Vorfeld* since *Vorfeld* material is systematically attached to the finite verb by the conversion procedure.
3. Extraposed material is dependent on its antecedent [12, pp. 130][23, pp. 101].

. In the remaining sentences, there are two phenomena which give rise to ill-nested structures, namely coordinated structures and discontinuous subjects. Fig. 7 shows the annotation of (2), an example from TIGER-Dep for a discontinuous subject. Again, the relevant edges are dashed.

(2) Auch würden durch die Regelung nur ständig neue Altfälle entstehen.
Also would through the regulation only always new cases emerge
"Another effect of the regulation would be constantly emerging new cases."

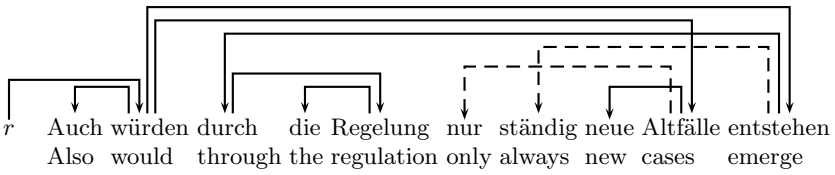

Fig. 7. Ill-nestedness due to a discontinuous subject

The ill-nested annotation of the coordination cases is largely disputable; however, this can be explained with the lack of a general linguistic theory of coordination

[25]. The situation for discontinuous subjects is clearer, since one can argue that the components of the discontinuous subject distinguish themselves from the material in the gap by making up a semantic unit.

To sum up, ill-nested dependency annotation in NeGra-Dep and TIGER-Dep can generally be linguistically justified and is not due to annotation oddities. An accurate linguistic survey of (k)-ill-nestedness of structures in DDT and PDT has not been presented in the literature and is left for future work.

4.2 Discontinuous Constituent Structures

In the following, we investigate gap degree and well-nestedness of constituent treebanks in order to verify if both measures are as informative for constituent trees as they are for dependency trees. We conduct our study on the constituent versions of the treebanks in the previous section, using exactly the same set of sentences (with removed punctuation). The quantitative results are summarized in Fig. 2.

	NeGra		TIGER	
total	20597		40013	
gap degree 0	14,648	72.44%	28,414	71.01%
gap degree 1	5,253	24.23%	10,310	25.77%
gap degree 2	687	3.30%	1,274	3.18%
gap degree 3	9	0.04%	15	0.04%
well-nested	20339	98.75%	39573	98.90%
1-ill-nested	258	1.25%	440	1.10%

Table 2. Constituent gap degree of TIGER/NeGra trees

The constituent gap degree figure of both German treebanks again lie close together. We found that the number of constituent structures with gap degree ≥ 3 is considerably lower than the corresponding number of dependency structures. The reason is that the phenomena which cause a high gap degree in dependency structures (enumerations and appositions) generally receive a constituent structure without gaps. The most frequent reasons for gaps in constituent structures are parenthetical constructions, as well as finite verbs, subjects and negative markers, which are generally annotated as immediate constituents of the highest S node and therefore may cause gaps in the VP yield.

Tab. 2 shows that the ratio of ill-nested structures in constituent data is comparable to the ratio in dependency data. This suggests that ill-nestedness has a comparable explanatory value as a constraining feature for constituent structures. As in the dependency treebanks, the only degree of ill-nestedness that can be observed is 1-ill-nestedness. A linguistic inspection of the ill-nested constituent structures shows that most of them are ill-nested due to the interplay of several annotation principles. Again, most of the ill-nested constituent

structures in TIGER and NeGra arise from extraposition phenomena. Furthermore we can also find cases of discontinuous subjects annotated with ill-nested structures. Other than for the dependency structures, coordination is no trigger for ill-nestedness in the constituent data.

As an example for ill-nested constituent structure, see the embedded sentence (3) and its tree annotation in Fig. 8.

(3) ...ob auf deren Gelände der Typ von Abstellanlage gebaut
...whether on their premises the type of parking facility built
werden könne, der ...
be could, which ...
"whether on their premises precisely the type of parking facility could be built, which ..."

The two overlapping, disjunctive constituents are the lower VP, and the NP with its extraposed relative clause.

The following underlying annotation principles seem to be respected throughout:

1. the subject is an immediate constituent of the sentence;
2. the finite verb is another immediate constituent (and the head) of the sentence;
3. the non-finite verb is the head of another immediate constituent that also includes objects and modifying expressions;
4. extraposed material is included in the antecedent constituent.

We will not argue in favor or against these annotation principles from a linguistic point of view. A mapping to common linguistic theories is highly non-trivial, since very different means of expressing constituency relations and a variety of shapings of constituent structures would have to be taken into account. On the other hand, the similarity to the above stated annotation principles for dependency structures is striking.

Ill-nestedness does not affect the same set of structures across treebank variants, i.e., the ill-nested dependency structures are no subset of the ill-nested constituent structures or vice versa. We leave for future work an exhaustive investigation of the differences.

5 Conclusion

We have presented a formal account for two measures of discontinuity, resp. non-projectivity, namely gap degree and well-nestedness. Our definitions emphasize that the notions of gaps and well-nestedness are independent of the type of syntactic structures, since they apply to both dependency and constituent structures. Concerning their application on dependency structures, they correspond to the well-known definitions from the literature. We have conducted an empirical study on two different versions of two treebanks, one annotated with constituents

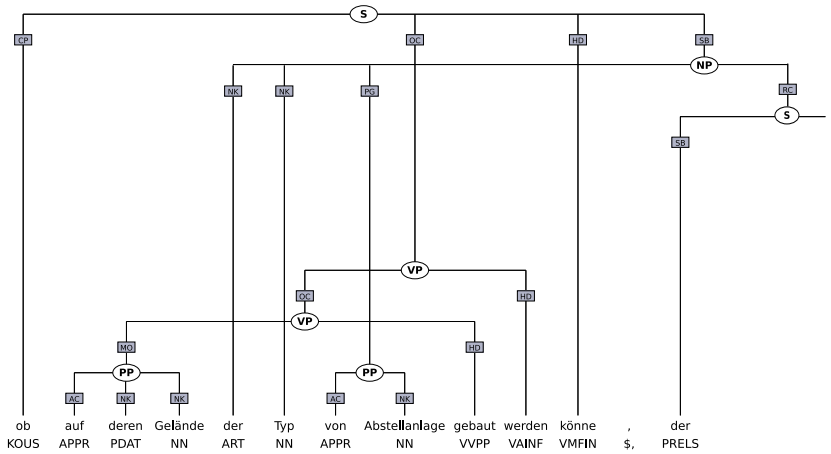

Fig. 8. An ill-nested sentence from NeGra

with crossing branches, the other one annotated with non-projective dependencies. The results show that the explanatory value of our measures applied on constituent trees is as high as for dependency structures. Even though well-nestedness has an almost perfect coverage on both constituent and dependency data, a linguistic inspection of ill-nested dependency and constituent structures shows that there exist linguistic phenomena for which ill-nested can indeed by linguistically justified.

As mentioned before, dependency conversion procedures introduce undesired noise. Therefore, in future work, we plan to undertake an exhaustive investigation of the exact effects of our converter on critical examples. Furthermore, we plan to conduct a study on Czech and Danish constituent data[3] and on Bulgarian constituent and dependency data [26], as well as an analysis of the conversion methods for dependency treebanks.

References

1. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of English: The Penn Treebank. Computational Linguistics **19**(2) (1994) 313–330
2. Civit, M., Martí Antònín, M.A.: Design principles for a Spanish treebank. In: Proceedings of the 1st Workshop on Treebanks and Linguistic Theories, Sozopol, Bulgaria (2002)
3. Telljohann, H., Hinrichs, E., Kübler, S., Zinsmeister, H.: Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z). Technischer Bericht, Seminar für Sprachwissenschaft, Universität Tübingen, Tübingen (July 2006) Revidierte Fassung.

[3] To our knowledge, such data does not exist and would have to be generated from the dependency data first.

4. Skut, W., Krenn, B., Brants, T., Uszkoreit, H.: An annotation scheme for free word order languages. In: Proceedings of the 5th Applied Natural Language Processing Conference, Washington, DC (1997) 88–95
5. Brants, S., Dipper, S., Hansen, S., Lezius, W., Smith, G.: The TIGER Treebank. In: Proceedings of the 1st Workshop on Treebanks and Linguistic Theories, Sozopol, Bulgaria (2002) 24–42
6. Kübler, S., Hinrichs, E.W., Maier, W.: Is it really that difficult to parse German? In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia (July 2006) 111–119
7. Boyd, A.: Discontinuity revisited: An improved conversion to context-free representations. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, the Linguistic Annotation Workshop, Prague, Czech Republic (2007) 41–44
8. Kuhlmann, M.: Dependency Structures and Lexicalized Grammars. PhD thesis, Saarland University (2007)
9. Holan, T., Kuboň, V., Oliva, K., Plátek, M.: Two useful measures of word order complexity. In: Workshop on Processing of Dependency-Based Grammars, Montréal, Canada (1998) 21–29
10. Bodirsky, M., Kuhlmann, M., Möhl, M.: Well-nested drawings as models of syntactic structure. In: Proceedings of the 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language (FG-MOL05), Edinburgh, UK (2005)
11. Kuhlmann, M., Satta, G.: Treebank grammar techniques for non-projective dependency parsing. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, Athens, Greece (2009)
12. Kunze, J.: Abhängigkeitsgrammatik. Volume 12 of Studia grammatica. Akademie-Verlag, Berlin (1975)
13. Havelka, J.: Beyond projectivity: Multilingual evaluation of constraints and measures on non-projective structures. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics. (2007) 608–615
14. Kuhlmann, M., Nivre, J.: Mildly non-projective dependency structures. In: Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, Sydney, Australia (2006)
15. Gómez-Rodríguez, C., Weir, D., Carroll, J.: Parsing mildly non-projective dependency structures. In: Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), Athens, Greece, Association for Computational Linguistics (March 2009) 291–299
16. Vijay-Shanker, K., Weir, D., Joshi, A.: Characterising structural descriptions used by various formalisms. In: Proceedings of ACL. (1987)
17. Boullier, P.: Proposal for a natural language processing syntactic backbone. Rapport de Recherche RR-3342, Institut National de Recherche en Informatique et en Automatique, Le Chesnay, France (1998)
18. Maier, W., Søgaard, A.: Treebanks and mild context-sensitivity. In: Proceedings of the 13th Conference on Formal Grammar 2008, Hamburg, Germany (2008) 61–76
19. Hajič, J., Hladka, B.V., Panevová, J., Hajičová, E., Sgall, P., Pajas, P.: Prague Dependency Treebank 1.0. LDC (2001) 2001T10.
20. Kromann, M.T.: The Danish Dependency Treebank and the DTAG treebank tool. In: Second Workshop on Treebanks and Linguistic Theories, Växjö, Sweden (2003) 217–220
21. Daum, M., Foth, K., Menzel, W.: Automatic transformation of phrase treebanks to dependency trees. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon, Portugal (2004)

22. Forst, M., Bertomeu, N., Crysmann, B., Fouvry, F., Hansen-Schirra, S., , Kordoni, V.: Towards a dependency-based gold standard for German parsers: The TiGer Dependency Bank. In: Proceedings of LINC 2004, Geneva, Switzerland (2004)
23. Hudson, R.: Word Grammar. Basil Blackwell, Oxford (1984)
24. Engel, U.: Deutsche Grammatik. Groos, Heidelberg (1988)
25. Lobin, H.: Koordinationssyntax als prozedurales Phänomen. Volume 46 of Studien zur deutschen Grammatik. Narr, Tübingen (1993)
26. Osenova, P., Simov, K.: BTB-TR05: BulTreebank Stylebook. Technical Report 05, BulTreeBank Project (2004)