Towards a Syntax-Semantics Interface for Topological Dependency Grammar

Ralph Debusmann and Denys Duchier

We present the first step towards a constraint-based syntax-semantics interface for Topological Dependency Grammar (TDG) (Duchier & Debusmann 2001). We extend TDG with a new level of representation called *semantic dependency dag* to capture the semantic dependencies and clearly separate this level from the *syntactic dependency tree*. We stipulate an emancipation mechanism between these levels that relates semantic arguments to their syntactic realizations, and demonstrate its application with an elegant account of raising and control constructions.

1 Introduction

(Duchier & Debusmann 2001) introduced the dependency-based grammar formalism of Topological Dependency Grammar (TDG), to account for the challenging word order phenomena in freer word order languages such as German. TDG explains linearization phenomena through the interaction of two structures, similar to (Gerdes & Kahane 2001): a non-ordered tree of syntactic dependencies, where edges are labeled by grammatical functions, and an ordered and projective tree of topological dependencies, where edges are labeled by topological fields.

In constrast to a *multi-stratal* approach such as MTT (Mel'čuk 1988), TDG is more properly said to be *multi-dimensional*. In MTT, the various levels of representation are organized vertically in a functional progression from one stratum to the next. In TDG, however, the various levels are organized horizontally and engage in constraint-based concurrent interactions.

Where (Duchier & Debusmann 2001) only attended to issues of syntax and linearization, we now broaden the scope and aim to equip TDG with a constraint-based syntax-semantics interface. Figure 1 illustrates the extended architecture where an analysis leads to the elaboration of a semantic representation.[1] In addition to the previous dimensions dedicated to syntactic

[1] In the diagram, we display an underspecified semantic representation using the Constraint Language for Lambda Structures (CLLS) (Egg, Koller & Niehren 2001).

dependencies (syntax) and word order (topology), we now postulate a new structure called *semantic dependency dag* to represent dependencies on semantic arguments; these dependencies provide us e.g. with the lambda-binding information required for semantics construction.

As in (Duchier & Debusmann 2001), the levels of syntax and topology are related through an emancipation mechanism which allows a word to *climb up* and *land* in the topological domain of a syntactic ancestor. Similarly, we now require the semantic dependency dag to be related to the syntactic dependency tree through an emancipation mechanism that allows a semantic argument to climb and be *realized* higher up in the syntax tree. In this article, we focus on this mechanism and demonstrate how the analysis of fairly complex control and raising constructions emerges from the constraint-based framework.

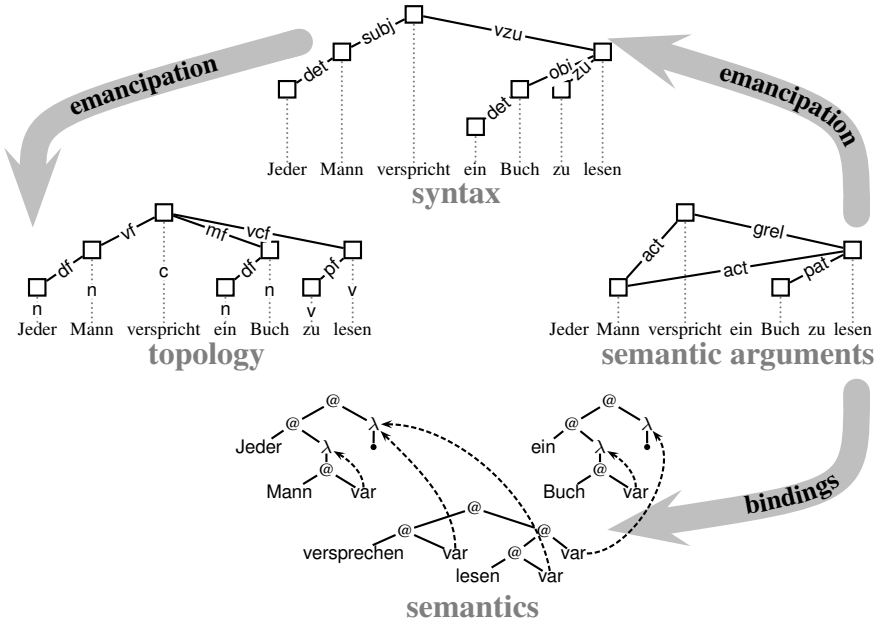


Figure 1: An architectural overview of TDG

2 Phenomena

In this section, we introduce the phenomena dealt with in this paper.

2.1 Raising

We start out with an example of raising:

Mary seems to laugh. (1)

Here, on the syntactic level, *seems* has a subject but *laugh* has none, while in the semantic argument structure, *laugh* has a deep subject or *actor* (*Mary*) but *seems* does not. We say that

the actor of the embedded verb *laugh* is realized as the subject of the raising verb *seems*. This phenomenon is called *subject-to-subject raising*.

If the actor of an embedded verb is realized by a direct object, we speak of subject-to-object raising. For example:

Mary believes him to laugh. (2)

Here, the direct object *him* of *believes* realizes the actor of *laugh*.

2.2 Control

Control verbs also realize the actor of an embedded verb as one of their complements. Contrary to raising verbs though, they assign this word an additional semantic role by themselves:

Mary tries to laugh. (3)

The subject *Mary* of *tries* is not only the actor of *laugh*, but also of the control verb *tries* itself. In other words, *Mary* fills two semantic roles at the same time. We call this phenomenon *subject-to-subject control*.

If the actor of an embedded verb is realized by a direct object, we speak of subject-to-object control:

Mary persuades him to laugh (4)

Here, the actor of *laugh* is realized as the object *him* of *persuades*.

3 TDG Framework

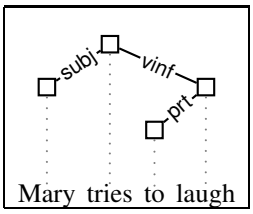
In this section, we provide an informal introduction to the TDG framework and describe our proposed extension. For a more theoretical presentation of TDG's formal foundations, we refer the reader to (Duchier 2001).

A TDG analysis consists of a lexical assignment and three structures: the syntactic dependency tree (ID for immediate dominance), the topological dependency tree (LP for linear precedence) and the semantic dependency dag (TH for thematic), which are formed from the same set of nodes (one for each word of the input) but different sets of edges. In this article, we ignore the LP tree and focus solely on the core of our proposal, namely the new TH dag and its relation to the ID tree.

3.1 Syntactic dependency tree

The ID tree is a non-ordered tree of syntactic dependencies where edges are labeled with grammatical functions such as subj for subject or obj for object. The ID tree-level closely corresponds to the analytical layer in FGD (Sgall, Hajicova & Panevova 1986), to the f-structure in LFG (Bresnan & Kaplan 1982) and to the DEPS-level in new versions of HPSG as e.g. in (Malouf 2000). Below, we show an example ID tree analysis of (5):

Mary tries to laugh. (5)

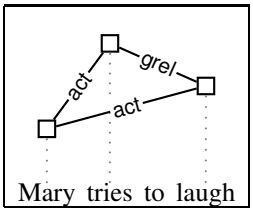


(6)

For this paper, we assume the set $\mathcal{G} = \{\mathsf{subj}, \mathsf{obj}, \mathsf{vinf}, \mathsf{prt}\}$ of grammatical functions, corresponding respectively to subject, object, *to*-infinitival complement and *to*-particle.

3.2 Semantic dependency dag

The TH dag is a directed acyclic graph of semantic dependencies. Like the ID tree, it is non-ordered and its edges are labeled by semantic roles such as act for actor (deep subject) and pat for patient (deep object). The TH dag-level closely corresponds to the tectogrammatical layer in FGD, to the a-structure in LFG, and to the ARG-ST in new versions of HPSG. Here is an example TH dag of sentence (5):



(7)

We do not commit ourselves to a particular set of semantic roles. For this article, we adopt a subset of the Praguian *dependency relations* developed for the dependency grammar formalism of FGD (Functional Generative Description) (Sgall et al. 1986):

$$\mathcal{S} = \{\mathsf{act}, \mathsf{pat}, \mathsf{grel}\}$$

act denotes the *actor* (deep subject), pat the *patient* (deep object) and grel a *general relationship*. We assign the latter to the predicates embedded under control and raising verbs as we could not find a suitable dependency relation for those predicates in the FGD-literature.

3.3 Lexical constraints

A TDG analysis is constrained by an assignment of lexical entries to nodes. A lexical entry has the signature:

$$\begin{bmatrix} \mathsf{in}_{\mathrm{ID}} & : & 2^{\mathcal{G}} \\ \mathsf{out}_{\mathrm{ID}} & : & 2^{\Pi_{\mathcal{G}}} \\ \mathsf{in}_{\mathrm{TH}} & : & 2^{\mathcal{S}} \\ \mathsf{out}_{\mathrm{TH}} & : & 2^{\Pi_{\mathcal{S}}} \\ \mathsf{link}_{\mathrm{TH}} & : & \mathcal{S} \rightarrow 2^{\mathcal{G}} \\ \mathsf{raised}_{\mathrm{TH}} & : & 2^{\mathcal{G}} \end{bmatrix}$$

For example, the lexical entry for the infinitive *laugh* is:

$$\textit{laugh} \quad = \quad \left[\begin{array}{rcl} \mathsf{in}_{\text{ID}} & : & \{\mathsf{vinf}\} \\ \mathsf{out}_{\text{ID}} & : & \{\mathsf{prt}\} \\ \mathsf{in}_{\text{TH}} & : & \{\mathsf{grel}\} \\ \mathsf{out}_{\text{TH}} & : & \{\mathsf{act}\} \\ \mathsf{link}_{\text{TH}} & : & \left[\mathsf{act} \mapsto \{\mathsf{subj}\}\right] \\ \mathsf{raised}_{\text{TH}} & : & \emptyset \end{array}\right]$$

In the remainder of the section, we explain these features and state the principles according to which a lexical assignment simultaneously constrains the ID tree, the TH dag, and the emancipation relationship between them, and thus restricts the admissible analyses.

Incoming edge principle. *In every structure of the analysis, the incoming edge (if any) of a node must be licensed by the corresponding* in *feature.*

The in_{ID} and in_{TH} features license the incoming edge respectively in the ID tree and in the TH dag. Thus, in the ID tree, *laugh* only accepts an incoming edge labeled vinf; we say that *laugh* has grammatical function vinf. In the TH dag, *laugh* only accepts an incoming edge labeled grel; we say that *laugh* fills the semantic role grel.

Outgoing edges principle. *In every structure of the analysis, the outgoing edges of a node must satisfy in label and number the stipulation of the corresponding* out *feature.*

The out_{ID} and out_{TH} features provide this stipulation respectively for the ID tree and the TH dag. Thus, in the ID tree, *laugh* requires precisely one outgoing edge labeled prt for the *to*-particle and admits no other. In the TH dag, *laugh* requires one outgoing edge labeled act for the actor and no other.

The stipulation of an out feature is expressed as a set of *label patterns*. Given a set $\mathcal{L}$ of labels, we write $\Pi_{\mathcal{L}}$ for the set of label patterns π that can be formed according to the following abstract syntax:

$$\pi \quad ::= \quad \ell \mid \ell? \mid \ell* \qquad\qquad \forall \ell \in \mathcal{L}$$

These patterns are used to distinguish obligatory and optional complements: ℓ means precisely one edge labeled ℓ (obligatory), $\ell?$ means at most one (optional), and $\ell*$ means 0 or more.

Linking principle. *The semantic arguments of a node, i.e. its dependents in the* TH *dag, must be syntactically realized in the* ID *tree with grammatical functions stipulated in the* $\mathsf{link}_{\text{TH}}$ *feature.*

$\mathsf{link}_{\text{TH}}$ describes a mapping between semantic roles and sets of grammatical functions which may realize them. In the example lexical entry above, the actor of *laugh* (act) must be realized as a subject (subj). Implicitly, the other semantic roles are mapped to the empty set, i.e. they cannot be realized by any grammatical function. The $\mathsf{link}_{\text{TH}}$ feature and the linking principle are more thoroughly discussed in (Korthals & Debusmann 2002).

The remaining features concern the *emancipation principle* which covers both raising and control constructions and explains how a semantic argument of an embedded verb can be

realized as a syntactic dependent of a dominating raising or control verb. We illustrate it with a lexical entry for control verb *tries*:

$$tries \;=\; \left[\begin{array}{rcl} \mathsf{in}_{\mathrm{ID}} & : & \emptyset \\ \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{subj}?, \mathsf{vinf}\} \\ \mathsf{in}_{\mathrm{TH}} & : & \emptyset \\ \mathsf{out}_{\mathrm{TH}} & : & \{\mathsf{act}, \mathsf{grel}\} \\ \mathsf{link}_{\mathrm{TH}} & : & \left[\begin{array}{l} \mathsf{act} \mapsto \{\mathsf{subj}\} \\ \mathsf{grel} \mapsto \{\mathsf{vinf}\} \end{array}\right] \\ \mathsf{raised}_{\mathrm{TH}} & : & \{\mathsf{subj}\} \end{array}\right]$$

Emancipation principle. *(a) only subjects may emancipate. (b) an emancipated subject must be realized in a raising/control position. (c) a raising/control position must realize the emancipated subject of at least one embedded verb.*

Stipulation (a) states that a semantic argument of a word w must also be realized as a syntactic dependent of w, except if it is $\mathsf{link}_{\mathrm{TH}}$-mapped to subj in w's lexical entry, in which case it may emancipate and be realized higher up in the ID tree. For (b), the feature $\mathsf{raised}_{\mathrm{TH}}$ indicates available *raising/control positions*: thus subj is a control position for *tries*.

Note that for simplicity, we drop the feature $\mathsf{blocks}_{\mathrm{TH}}$ and the corresponding *barriers principle*. The barriers principle prohibits that nodes climb "too far up", e.g. that they should not climb through finite verbs.

3.4 Lexical inheritance

TDG is a highly lexicalized grammar formalism, and in order to express linguistic generalizations, we make use of a mechanism of *lexical inheritance*. This mechanism is thoroughly described in (Debusmann 2001) and allows us to compose *lexical entries* from a number of *lexical types* (prefixed with "*t_*") using lattice operations. For instance we obtain the lexical entry for *tries* (as given above) as follows:

$$tries \;=\; t\_finite \sqcap t\_grel\_vinf \sqcap t\_c\_subj\_to\_subj$$

where *t_finite* is the lexical type for finite verbs, *t_grel_vinf* for verbs whose infinitival complement realizes a general relationship (grel) and *t_c_subj_to_subj* for subject-to-subject-control verbs. Inheritance amounts to set intersection for features $\mathsf{in}_{\mathrm{ID}}$ and $\mathsf{in}_{\mathrm{TH}}$, and set union for $\mathsf{out}_{\mathrm{ID}}$, $\mathsf{out}_{\mathrm{TH}}$, $\mathsf{link}_{\mathrm{TH}}$ and $\mathsf{raised}_{\mathrm{TH}}$. Omitted features are assigned a *default value* (lattice top): the full set of labels for $\mathsf{in}_{\mathrm{ID}}$ and $\mathsf{in}_{\mathrm{TH}}$, and the empty set for all other features.

4 Grammar fragment

In this section, we present a grammar fragment covering the phenomena outlined in section 2. The grammar fragment mainly consists of a number of lexical types from which we can obtain the individual lexical entries by lexical inheritance.

Nouns. We begin with the lexical type for nouns:

$$t\_noun \;=\; \begin{bmatrix} \mathsf{in}_{\mathrm{ID}} & : & \{\mathsf{subj}, \mathsf{obj}\} \\ \mathsf{in}_{\mathrm{TH}} & : & \{\mathsf{act}, \mathsf{pat}\} \end{bmatrix}$$

That is: on the syntactic level, nouns may have either the grammatical function subject (subj) or object (obj), while on the semantic level, they may fill either the actor (act) or patient (pat) roles. We define expletives (e.g. *it*) as nouns which cannot fill a semantic role (i.e. their $\mathsf{in}_{\mathrm{TH}}$-set is empty).

Finite verbs. In this fragment, finite verbs are matrix verbs, thus have no incoming edges and an optional subject:

$$t\_finite \;=\; \begin{bmatrix} \mathsf{in}_{\mathrm{ID}} & : & \emptyset \\ \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{subj}?\} \\ \mathsf{in}_{\mathrm{TH}} & : & \emptyset \end{bmatrix}$$

Infinite verbs. For the small grammar fragment described in this article, we only consider *to*-infinitives requiring a *to*-particle (prt). Their incoming edge label must be vinf in the ID tree, and grel in the TH dag:

$$t\_infinite \;=\; \begin{bmatrix} \mathsf{in}_{\mathrm{ID}} & : & \{\mathsf{vinf}\} \\ \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{prt}\} \\ \mathsf{in}_{\mathrm{TH}} & : & \{\mathsf{grel}\} \end{bmatrix}$$

Linking types. *Linking types* describe how semantics roles can be realized by grammatical functions (Korthals & Debusmann 2002). For this fragment, we only define three linking types. The first (*t_act_subj*) realizes the actor by a subject:

$$t\_act\_subj \;=\; \begin{bmatrix} \mathsf{out}_{\mathrm{TH}} & : & \{\mathsf{act}\} \\ \mathsf{link}_{\mathrm{TH}} & : & [\mathsf{act} \mapsto \{\mathsf{subj}\}] \end{bmatrix}$$

The second (*t_pat_obj*) realizes the patient by an object:

$$t\_pat\_obj \;=\; \begin{bmatrix} \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{obj}\} \\ \mathsf{out}_{\mathrm{TH}} & : & \{\mathsf{pat}\} \\ \mathsf{link}_{\mathrm{TH}} & : & [\mathsf{pat} \mapsto \{\mathsf{obj}\}] \end{bmatrix}$$

The third (*t_grel_vinf*) states that the embedded predicate of a verb is assigned the semantic role grel and must be realized by an infinitive:

$$t\_grel\_vinf \;=\; \begin{bmatrix} \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{vinf}\} \\ \mathsf{out}_{\mathrm{TH}} & : & \{\mathsf{grel}\} \\ \mathsf{link}_{\mathrm{TH}} & : & [\mathsf{grel} \mapsto \{\mathsf{vinf}\}] \end{bmatrix}$$

Raising. The raising-position for subject-to-subject raising verbs is subj:

$$t\_r\_subj\_to\_subj \;=\; \left[\; \mathsf{raised}_{\mathrm{TH}} \;:\; \{\mathsf{subj}\} \;\right]$$

The actor of a subject-to-object raising verb is realized by a subject, and its raising-position is obj:

$$t\_r\_subj\_to\_obj \;=\; t\_act\_subj \sqcap \left[\begin{array}{rcl} \mathsf{out}_{\mathrm{ID}} & : & \{\mathsf{obj}\} \\ \mathsf{raised}_{\mathrm{TH}} & : & \{\mathsf{obj}\} \end{array}\right]$$

Control. We model control as a special case of raising. Hence, the lexical type for subject-to-subject control verbs inherits from the lexical type for subject-to-subject raising verbs. Contrary to a raising verb, a control verb realizes its actor as a subject in addition:

$$t\_c\_subj\_to\_subj \;=\; t\_r\_subj\_to\_subj \sqcap t\_act\_subj$$

We model subject-to-object control verbs as subject-to-object raising verbs which require a patient realized as their object:

$$t\_c\_subj\_to\_obj \;=\; t\_r\_subj\_to\_obj \sqcap t\_pat\_obj$$

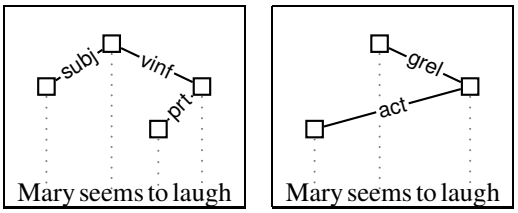
5 Application

In this section, we apply the TDG framework and the grammar fragment outlined above to the phenomena laid out in section 2.

5.1 Raising

We begin with a subject-to-subject raising example and the corresponding ID tree and TH dag analyses:

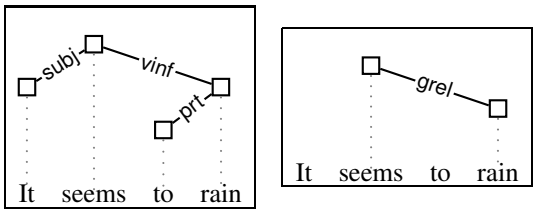
Mary seems to laugh. (8)



Here, the actor *Mary* of *laugh* has climbed up (or emancipated) and is syntactically realized as the subject of the raising verb *seems*. The latter assigns no semantic role to its subject (i.e. there is no edge from *seems* to *Mary* in the TH dag). The *to*-particle fills no semantic role; it is isolated in the TH dag.

Now consider the sentence:

It(expl) seems to rain. (9)



Here, no emancipation takes place since the raising verb *seems* embeds a verb without an actor (*rain*). The expletive *it* fills no semantic role and is thus isolated in the TH dag.

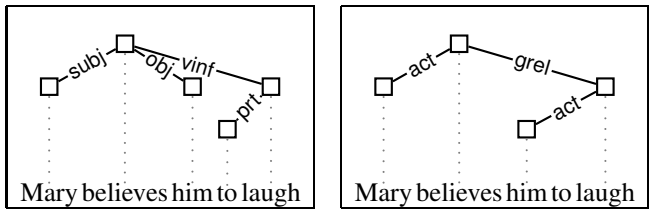
We turn to the following ungrammatical sentence:

$$*\textit{Mary seems to rain} \tag{10}$$

Here, the emancipation principle (c) is violated that requires a raising position (here *Mary*) to realize at least one emancipated subject. But *rain* has no actor which could emancipate.

We turn to an example of subject-to-object raising:

$$\textit{Mary believes him to laugh.} \tag{11}$$



This time, the actor of the embedded verb *laugh* climbs up and is realized as the object *him* of the raising verb *believes*.

5.2 Control

Next, we discuss an example of subject-to-subject control:

$$\textit{Mary tries to laugh.} \tag{12}$$

The ID tree and the TH dag analyses of the sentence are provided in (6) and (7) above. There, the actor *Mary* of the embedded verb *laugh* emancipates and is realized as the subject of the control verb *tries*. Contrary to a raising verb though, the control verb *tries* also assigns a semantic role (actor) to the emancipated subject, i.e. *Mary* fills a semantic role on two verbs at the same time: it is the actor of *tries* and the actor of *laugh*. In the TH dag, this is reflected by two incoming edges to *Mary*.

Finally, control verbs cannot embed verbs without an actor:

$$*\textit{It(expl) tries to rain} \tag{13}$$

In our framework, this sentence is not licensed because *tries* requires an actor on the TH-level but does not get one. Thus, the outgoing edges principle is violated.

6 Conclusion

We presented the first steps towards a constraint-based syntax-semantics interface for the TDG grammar formalism. We extended TDG with a new level of semantic dependencies (TH dag) which is clearly separated from the purely syntactic dependencies captured in the ID tree. The two levels interact through lexicalized constraints and principles, in particular, the emancipation principle. We illustrated how fairly complex phenomena, such as control and raising, can be modeled as emerging from the interactions of simple constraints. We have implemented a prototype constraint-based parser including the new semantic dependency dag-level which performs very well.

Coming back to the TDG architecture displayed in Figure 1, we now have the means to make the final transition to semantics. To this end, we plan to utilize the information contained in the semantic argument dag to obtain a description of an underspecified semantics using the Constraint Language for Lambda Structures (CLLS) (Egg et al. 2001).

References

Bresnan, J. & Kaplan, R. (1982), Lexical-functional grammar: A formal system for grammatical representation, *in* 'The Mental Representation of Grammatical Relations', MIT Press.

Debusmann, R. (2001), A declarative grammar formalism for dependency grammar, Master's thesis, University of Saarland.

Duchier, D. (2001), Lexicalized syntax and topology for non-projective dependency grammar, *in* 'MOL 8 Proceedings'.

Duchier, D. & Debusmann, R. (2001), Topological dependency trees: A constraint-based account of linear precedence, *in* 'ACL 2001 Proceedings'.

Egg, M., Koller, A. & Niehren, J. (2001), 'The constraint language for lambda structures', *Journal of Logic, Language, and Information* .

Gerdes, K. & Kahane, S. (2001), Word order in german: A formal dependency grammar using a topological hierarchy, *in* 'ACL 2001 Proceedings'.

Korthals, C. & Debusmann, R. (2002), Linking syntactic and semantic arguments in a dependency-based formalism, *in* 'COLING 2002 Proceedings'.

Malouf, R. (2000), A head-driven account of long-distance case assignment, *in* 'Grammatical Interfaces in HPSG', CSLI Publications.

Mel'čuk, I. (1988), *Dependency Syntax: Theory and Practice*, State Univ. Press of New York, Albany/NY.

Sgall, P., Hajicova, E. & Panevova, J. (1986), *The Meaning of the Sentence in its Semantic and Pragmatic Aspects*, D. Reidel.