

# Constraint-based Computational Semantics: A Comparison between LTAG and LRS

**Laura Kallmeyer**

University of Tübingen  
Collaborative Research Center 441  
lk@sfs.uni-tuebingen.de

**Frank Richter**

University of Tübingen  
Collaborative Research Center 441  
fr@sfs.uni-tuebingen.de

## Abstract

This paper compares two approaches to computational semantics, namely semantic unification in Lexicalized Tree Adjoining Grammars (LTAG) and Lexical Resource Semantics (LRS) in HPSG. There are striking similarities between the frameworks that make them comparable in many respects. We will exemplify the differences and similarities by looking at several phenomena. We will show, first of all, that many intuitions about the mechanisms of semantic computations can be implemented in similar ways in both frameworks. Secondly, we will identify some aspects in which the frameworks intrinsically differ due to more general differences between the approaches to formal grammar adopted by LTAG and HPSG.

## 1 Introduction

This paper contrasts two frameworks for computational semantics, the proposal for semantics in LTAG described in (Kallmeyer and Romero, 2005) and LRS (Richter and Sailer, 2004), a computational semantics framework formulated in Head-Driven Phrase Structure Grammar (HPSG).

There are significant differences between LTAG and HPSG. LTAG is a mildly context-sensitive lexicalized formalism characterized by an extended domain of locality. HPSG is based on the idea of a separation of the lexicon and syntactic structure and on the strict locality of general grammar principles that are formulated in an expressive and very flexible logical description language. These fundamental differences are reflected in the respective architectures for semantics: LTAG assumes a separate level of underspecified semantic

representations; LRS uses the description logic of syntax for semantic specifications.

However, despite the different mathematical structures, we find striking similarities between LTAG semantics with unification and LRS. They both show similar intuitions underlying specific analyses, use the same higher order type-theoretic language (Ty2, (Gallin, 1975)) as a means for specifying the truth conditions of sentences, and employ a feature logic in the combinatorial semantics instead of the lambda calculus. Because of these similarities, analyses using both approaches are closely related and can benefit from each other.

The paper is structured as follows: Sections 2 and 3 will introduce the two frameworks. The next three sections (4–6) will sketch analyses of some phenomena in both frameworks that will reveal relevant relations between them. Section 7 presents a summary and conclusion.

## 2 LTAG semantics

In (Kallmeyer and Romero, 2005), each elementary tree is linked to a semantic representation (a set of Ty2 formulas and scope constraints). Ty2 formulas (Gallin, 1975) are typed  $\lambda$ -terms with individuals and situations as basic types. The scope constraints of the form  $x \geq y$  specify subordination relations between Ty2 terms. In other words,  $x \geq y$  indicates that  $y$  is a component of  $x$ .

A semantic representation is equipped with a semantic feature structure description. Semantic computation is done on the derivation tree and consists of certain feature value equations between mother and daughter nodes in the derivation tree.

(1) John always laughs.

As an example, see Fig. 1 showing the derivation tree for (1) with semantic representations and

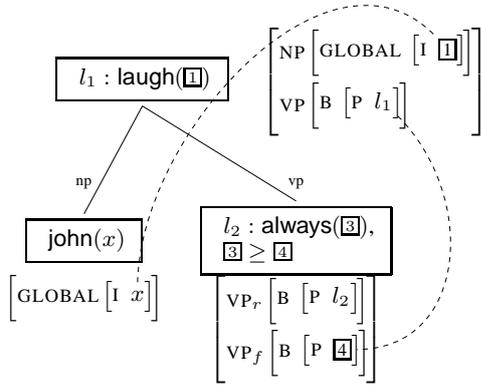


Figure 1: LTAG semantics of (1)

semantic feature structure descriptions as node labels. The additional feature equations in this example are depicted using dotted lines. They arise from top-bottom feature identifications parallel to the unifications performed in FTAG (Vijay-Shanker and Joshi, 1988) and from identifications of global features. They yield  $\boxed{1} = x$  and  $\boxed{4} = l_1$ . Applying these identities to the semantic representations after having built their union leads to (2). The constraint  $\boxed{3} \geq l_1$  states that  $l_1 : \text{laugh}(x)$  is a component of  $\boxed{3}$ .

$$(2) \quad \boxed{\text{john}(x), l_2 : \text{always}(\boxed{3}), l_1 : \text{laugh}(x), \boxed{3} \geq l_1}$$

Note that the feature structure descriptions do not encode the semantic expressions one is interested in. They only encode their contributions to functional applications by restricting the argument slots of certain predicates in the semantic representations: They state which elements are contributed as possible arguments for other semantic expressions and which arguments need to be filled. They thereby simulate lambda abstraction and functional application while assembling the semantic representations. To achieve this, a restricted first order logic is sufficient.

Semantic computation is local on the derivation tree: The new feature equations that are added depend only on single edges in the derivation tree. Because of this, even with the extension to semantics, the formalism is still mildly context-sensitive.

### 3 LRS

In LRS the feature logic specifies the entire grammar, including well-formed Ty2 terms as semantic representations, and their mode of composition. Instead of the lambda calculus of traditional Montague Grammar, LRS crucially uses a

novel distinction between three aspects of the logical representations of signs (external content, internal content, and parts). LRS constraints establish sub-term relationships between pieces of semantic representations within and across signs, thereby specifying the combinatorial properties of the semantics. The subterm or component-of conditions (symbolized as  $\triangleleft$ ) are imposed by grammar principles. Since these principles are descriptions of object-language expressions, they permit the application of various underspecification techniques of computational semantics, although an LRS grammar does not employ underspecified semantic representations, in contrast to LTAG semantics.

Fig. 2 shows an HPSG description of the syntactic tree and the LRS specifications of (1). The syntactic trees in HPSG correspond to the derived trees of LTAG. Since HPSG does not have derivation trees, the LRS principles refer to derived trees.

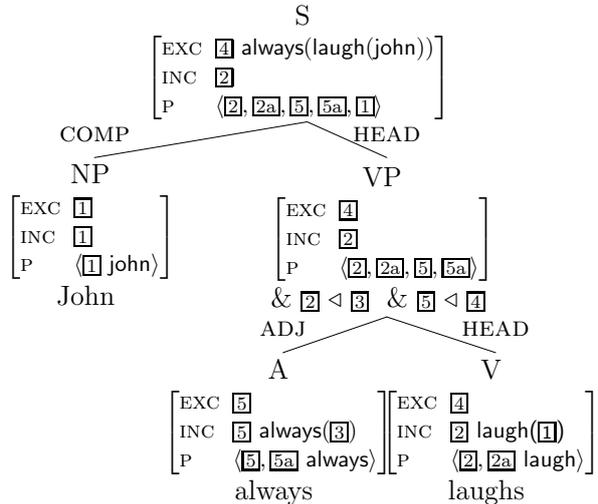


Figure 2: LRS analysis of (1)

Each word lexically specifies its contribution to the overall meaning of the sentence ( $P(\text{ARTS})$ ), the part of its semantics which is outscoped by all signs the word combines with ( $\text{INC}(\text{ONT})$ ), and the overall semantic contribution of its maximal projection ( $\text{EXC}(\text{ONT})$ ). Feature percolation principles identify  $\text{INC}$  and  $\text{EXC}$ , respectively, along head projections and collect the elements of the  $\text{PARTS}$  lists of the daughters at each phrase. The combination of the adjunct with a verbal projection introduces two component-of constraints: The  $\text{EXC}$  of *always* must be within the  $\text{EXC}$  of *laughs*, and the  $\text{INC}$  of *laughs* must be in the scope of *always*. The semantic argument of

*laughs* (john) is identified by subcategorization (not shown in Fig. 2). A closure condition requires that the semantic representation of an utterance use up all and only the PARTS contributions of all signs, which yields  $\boxed{4} = \text{always}(\text{laugh}(\text{john}))$ .

## 4 Quantifier scope

### 4.1 Specifying a scope window

- (3) Exactly one student admires every professor:  
 $\exists > \forall, \forall > \exists$
- (4) John seems to have visited everybody:  
 $\text{seem} > \forall, \forall > \text{seem}$

Quantificational NPs in English can in principle scope freely (see (3) and (4)). An analysis of quantifier scope must guarantee only two things: 1. the proposition to which a quantifier attaches must be in its nuclear scope, and 2. a quantifier cannot scope higher than the next finite clause. One way to model this is to define a scope window delimited by a maximal scope and a minimal scope for a quantifier. Both LTAG and LRS, specify such scope windows for quantifiers. We will now outline the two analyses.

- (5) Everybody laughs.

(Kallmeyer and Romero, 2005) use global features MAXS and MINS for the limits of the scope window. Fig. 3 shows the LTAG analysis of (5). The feature identifications (indicated by dotted lines) lead to the constraints  $\boxed{2} \geq \boxed{5}, \boxed{5} \geq l_1$ . These constraints specify an upper and a lower boundary for the nuclear scope  $\boxed{5}$ . With the assignments following from the feature identifications we obtain the semantic representation (6):

$$(6) \begin{array}{l} l_1 : \text{laugh}(x), \\ l_2 : \text{every}(x, \boxed{4}, \boxed{5}), l_3 : \text{person}(x) \\ \boxed{2} \geq l_1, \\ \boxed{4} \geq l_3, \boxed{2} \geq \boxed{5}, \boxed{5} \geq l_1 \end{array}$$

There is one possible disambiguation consistent with the scope constraints, namely  $\boxed{2} \rightarrow l_2, \boxed{4} \rightarrow l_3, \boxed{5} \rightarrow l_1$ . This leads to the semantics  $\text{every}(x, \text{person}(x), \text{laugh}(x))$ .

In LRS, the EXCONT value of the utterance is the upper boundary while the INCONT value of the syntactic head a quantifier depends on is the lower boundary for scope, as illustrated in Fig. 4. The upper boundary is obtained through the interaction of 1) a PROJECTION PRINCIPLE stating that the

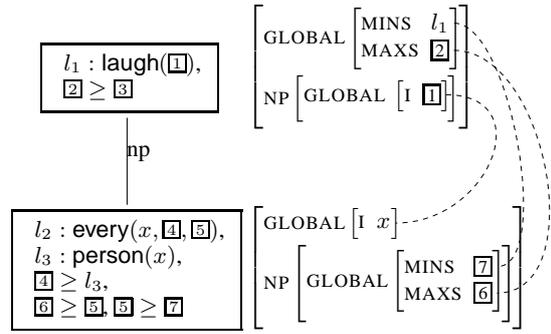
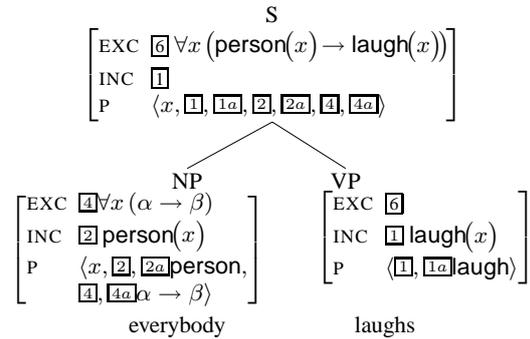


Figure 3: LTAG analysis of (5) *Everybody laughs*

PARTS list of a phrase contains all elements on the PARTS lists of its daughters, and 2) the EXCONT PRINCIPLE which states that a) the PARTS list of each non-head contains its own EXCONT, and b) in an utterance, everything on the PARTS list is a component of the EXCONT. This leads to the constraint  $\boxed{4} < \boxed{6}$  in Fig. 4, among others. The lower boundary is obtained from the SEMANTICS PRINCIPLE which states that if the non-head of a headed phrase is a quantifier, then the INCONT of the head is a component of its nuclear scope. This yields  $\boxed{1} < \beta$  in Fig. 4.



Relevant subterm constraints:  $\boxed{2} < \alpha$  (from the lexical entry of *everybody*),  $\boxed{1} < \beta, \boxed{4} < \boxed{6}$

Figure 4: LRS analysis of (5) *Everybody laughs*

The striking similarity between the two analyses shows that, despite the fundamental differences between the frameworks, central insights can be modelled in parallel.

### 4.2 Nested quantifiers

The use of the upper limit of the scope windows is, however, slightly different: EXCONT contains the quantifier itself as a component while MAXS limits only the nuclear scope, not the quantifier. Consequently, in LTAG the quantifier can scope higher

than the MAXS limiting its nuclear scope but in this case it takes immediate scope over the MAXS.

- (7) Two policemen spy on someone from every city:  $\forall > \exists > 2$  (among others)

The LTAG analysis is motivated by nested quantifiers. In sentences such as (7), the embedded quantifier can take scope over the embedding one but if so, this must be immediate scope. In other words, other quantifiers cannot intervene. In (7), the scope order  $\forall > 2 > \exists$  is therefore not possible.<sup>1</sup> The LTAG analysis is such that the maximal nuclear scope of the embedded quantifier is the propositional label of the embedding quantifier.<sup>2</sup>

In LRS, the way the scope window is specified, a corresponding constraint using the EXCONT of the embedded quantifier cannot be obtained. The LRS principle governing the distribution of embedded quantifiers in complex NPs states directly that in this syntactic environment, the embedded quantifier may only take direct scope over the quantifier of the matrix NP. This principle does not refer to the notion of external content at all. At this point it is an open question whether LRS could learn from LTAG here and adapt the scope window so that an analogous treatment of nested quantifiers would be possible.

## 5 LTAG's extended domain of locality

Whereas the treatment of quantification sketched in the preceding section highlights the similarities between LTAG semantics and LRS, this and the following section will illustrate some fundamental differences between the frameworks.

In spite of the parallels mentioned above, even INCONT and MINS differ sometimes, namely in sentences containing bridge verbs. This is related to the fact that LTAG has an extended domain of locality whereas HPSG does not. Let us illustrate the difference with the example (8).

- (8) Mary thinks John will come.

<sup>1</sup>(Joshi et al., 2003) propose an extra mechanism that groups quantifiers into sets in order to derive these constraints. (Kallmeyer and Romero, 2005) however show that these constraints can be derived even if the upper limit MAXS for nuclear scope is used as sketched above.

<sup>2</sup>Note that this approach requires constraints of the form  $l \geq \boxed{v}$  with  $l$  being a label,  $\boxed{v}$  a variable. This goes beyond the polynomially solvable *normal dominance constraints* (Althaus et al., 2003). This extension, though, is probably still polynomially solvable (Alexander Koller, personal communication).

In LTAG, the two elementary verb trees (for *thinks* and *will come*) have different global MINS features. The one for *thinks* is the label of the think proposition while the one for *will come* is the label of the embedded proposition. As a consequence, a quantifier which attaches to the matrix verb cannot scope into the embedded clause. This distinction of different MINS values for different verb trees is natural in LTAG because of the extended domain of locality.

In LRS, all verbal nodes in the constituent structure of (8) carry the same INCONT value, namely the proposition of the embedded verb. Consequently, the minimal scope of quantifiers attaching either to the embedding or to the embedded verb is always the proposition of the embedded verb. However, due to the requirement that variables be bound, a quantifier binding an argument of the embedding verb cannot have narrow scope over the embedded proposition.

How to implement the LTAG idea of different INCONT values for the embedding and the embedded verb in LRS is not obvious. One might introduce a new principle changing the INCONT value at a bridge verb, whereby the new INCONT would get passed up, and the embedded INCONT would no longer be available. This would be problematic: Take a raising verb as in (9) (adjoining to the VP node in LTAG) instead of a bridge verb:

- (9) Most people seem to everybody to like the film.

Here the minimal scope of *most people* should be the *like* proposition while the minimal scope of *everybody* is the *seem* proposition. In LTAG this does not pose a problem since, due to the extended domain of locality, *most people* attaches to the elementary tree of *like* even though the *seem* tree is adjoined in between. If the INCONT treatment of LRS were modified as outlined above and *seem* had an INCONT value that differed from the INCONT value of the embedded *like* proposition, then the new INCONT value would be passed up and incorrectly provide the minimal scope of *most people*. LRS must identify the two INCONTs.

The difference between the two analyses illustrates the relevance of LTAG's extended domain of locality not only for syntax but also for semantics.

## 6 Negative Concord

The analysis of negative concord in Polish described in this section highlights the differences

in the respective implementation of underspecification techniques in LTAG and LRS. Recall that both LTAG and LRS use component-of constraints. But in LTAG, these constraints link actual Ty2-terms (i.e., objects) to each other, while in LRS, these constraints are part of a description of Ty2-terms.

- (10) Janek nie pomaga ojcu.  
 Janek NM helps father  
 ‘Janek doesn’t help his father.’
- (11) a. Janek nie pomaga nikomu.  
 Janek NM helps nobody  
 ‘Janek doesn’t help anybody.’
- b. \*Janek pomaga nikomu.

- (12) Nikt nie przyszedł.  
 nobody NM came  
 ‘Nobody came.’

The basic facts of sentential negation and negative concord in Polish are illustrated in (10)–(12): The verbal prefix *nie* is obligatory for sentential negation, and it can co-occur with any number of n-words (such as *nikt*, ‘anybody’) without ever leading to a double negation reading. As a consequence, (12) expresses only one logical sentential negation, although the negation prefix *nie* on the verb and the n-word *nikt* can carry logical negation alone in other contexts. LRS takes advantage of the fact that its specifications of semantic representations are descriptions of logical expressions which can, in principle, mention the same parts of the expressions several times. Fig. 5 shows that both *nikt* and the verb *nie przyszedł* introduce descriptions of negations (4 and 2, respectively). The constraints of negative concord in Polish will then conspire to force the negations contributed by the two words to be the same in the overall logical representation 6 of the sentence.

Such an analysis is not possible in LTAG. Each negation in the interpretation corresponds to exactly one negated term introduced in the semantic representations. Therefore, the negative particle *nie* necessarily introduces the negation while the n-word *nikt* requires a negation in the proposition it attaches to. An analysis along these lines is sketched in Fig. 6 (“GL” stands for “GLOBAL”). The requirement of a negation is checked with a feature NEG indicating the presence of a negation. The scope of the negation (feature N-SCOPE)

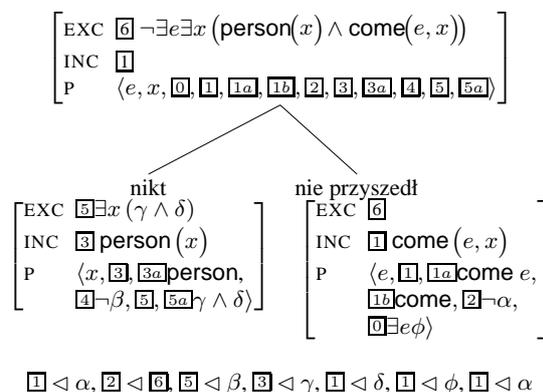


Figure 5: LRS analysis of (12) *Nikt nie przyszedł*

marks the maximal scope of the existential quantifier of the n-word *nikt* (constraint 7 ≥ 6).<sup>3</sup>

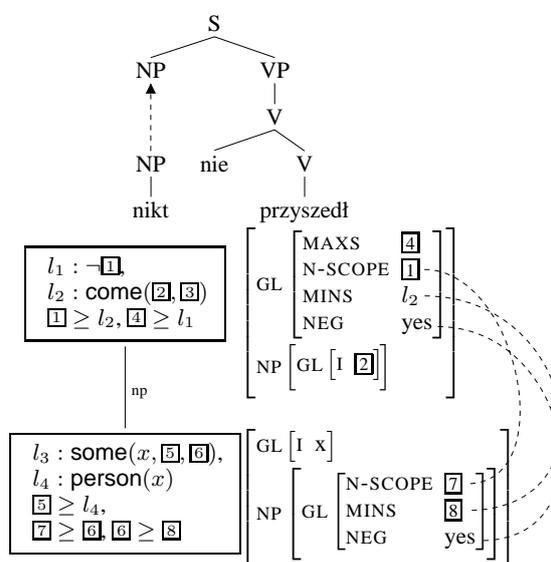


Figure 6: LTAG analysis of (12) *Nikt nie przyszedł*

This example illustrates that the two frameworks differ substantially in their treatment of underspecification: 1. LRS employs partial descriptions of fully specified models, whereas LTAG generates underspecified representations in the style of (Bos, 1995) that require the definition of a disambiguation (a “plugging” in the terminology of Bos). 2. LRS constraints contain not Ty2 terms but descriptions of Ty2 terms. Therefore, in contrast to LTAG, two descriptions can denote the same formula. Here, LTAG is more limited compared to LRS. On the other hand, the way semantic representations are defined in LTAG guarantees

<sup>3</sup>See (Lichte and Kallmeyer, 2006) for a discussion of NEG and N-SCOPE in the context of NPI-licensing.

that they almost correspond to *normal dominance constraints*, which are known to be polynomially parsable. The difference in the use of underspecification techniques reflects the more general difference between a generative rewriting system such as LTAG, in which the elements of the grammar are objects, and a purely description-based formalism such as HPSG, in which token identities between different components of linguistic structures are natural and frequently employed.

## 7 Summary and Conclusion

LTAG and LRS have several common characteristics: They both 1. use a Ty2 language for semantics; 2. allow underspecification (LTAG scope constraints  $\geq$  versus LRS component-of constraints  $\triangleleft$ ); 3. use logical descriptions for semantic computation; 4. are designed for computational applications. Due to these similarities, some analyses can be modelled in almost identical ways (e.g., the quantifier scope analyses, and the identification of arguments using attribute values rather than functional application in the lambda calculus). We take the existence of this clear correspondence as indicative of deeper underlying insight into the functioning of semantic composition in natural languages.

Additionally, the differences between the frameworks that can be observed on the level of syntax carry over to semantics: 1. LTAG's extended domain of locality allows the localization within elementary trees of syntactic and semantic relations between elements far apart from each other on the level of constituent structure. 2. LTAG (both syntax and semantics) is a formalism with restricted expressive power that guarantees good formal properties. The restrictions, however, can be problematic. Some phenomena can be more easily described in a system such as HPSG and LRS while their description is less straightforward, perhaps more difficult or even impossible within LTAG. The concord phenomena described in section 7 are an example of this.

A further noticeable difference is that within the (Kallmeyer and Romero, 2005) framework, the derivation tree uniquely determines both syntactic and semantic composition in a context-free way. Therefore LTAG semantics is mildly context-sensitive and can be said to be compositional. As far as LRS is concerned, it is not yet known whether it is compositional or not; compositional-

ity (if it holds at all) is at least less straightforward to show than in LTAG.

In conclusion, we would like to say that the similarities between these two frameworks permit a detailed and direct comparison. Our comparative study has shed some light on the impact of the different characteristic properties of our frameworks on concrete semantic analyses.

## Acknowledgments

For many long and fruitful discussions of various aspects of LTAG semantics and LRS, we would like to thank Timm Lichte, Wolfgang Maier, Maribel Romero, Manfred Sailer and Jan-Philipp Söhn. Furthermore, we are grateful to three anonymous reviewers for helpful comments.

## References

- Ernst Althaus, Denys Duchier, Alexander Koller, Kurt Mehlhorn, Joachim Niehren, and Sven Thiel. 2003. An efficient graph algorithm for dominance constraints. *Journal of Algorithms*, 48(1):194–219.
- Johan Bos. 1995. Predicate logic unplugged. In Paul Dekker and Martin Stokhof, editors, *Proceedings of the 10th Amsterdam Colloquium*, pages 133–142.
- Daniel Gallin. 1975. *Intensional and Higher-Order Modal Logic with Applications to Montague Semantics*. North Holland mathematics studies 19. North-Holland Publ. Co., Amsterdam.
- Aravind K. Joshi, Laura Kallmeyer, and Maribel Romero. 2003. Flexible Composition in LTAG: Quantifier Scope and Inverse Linking. In Harry Bunt, Ielka van der Sluis, and Roser Morante, editors, *Proceedings of the Fifth International Workshop on Computational Semantics IWCS-5*, pages 179–194, Tilburg.
- Laura Kallmeyer and Maribel Romero. 2005. Scope and Situation Binding in LTAG using Semantic Unification. Submitted to *Research on Language and Computation*. 57 pages., December.
- Timm Lichte and Laura Kallmeyer. 2006. Licensing German Negative Polarity Items in LTAG. In *Proceedings of The Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+8)*, Sydney, Australia, July.
- Frank Richter and Manfred Sailer. 2004. Basic concepts of lexical resource semantics. In Arnold Beckmann and Norbert Preining, editors, *ESSLLI 2003 – Course Material I*, (= Collegium Logicum, 5), pages 87–143. Kurt Gödel Society, Wien.
- K. Vijay-Shanker and Aravind K. Joshi. 1988. Feature structures based tree adjoining grammar. In *Proceedings of COLING*, pages 714–719, Budapest.