Higher Order Recursion for Syntax-Semantics Interface in Constraint-Based Grammar

Roussanka Loukanova
Computational Linguistics, Uppsala University

Synopsis Work on grammar developments made it clear that linguistic adequateness of a computational syntactic theory is based on its potentials to be integrated with semantic representations of its constructs, i.e., with computational semantics. This realization has been inciting research and implementation work in the field of *Computational Syntax-Semantics Interface*. Since their origins, constraint-based grammar approaches, in particular HPSG, have taken representation of semantic content as an essential part of grammar, on a par with syntactic descriptions. However, while the work done within constraint-based grammar (CBG) frameworks is considerable, the lexical and phrasal components are predominantly syntactic per se, and are in need of more substantial integration with semantic representations based on rigorous, and adequately expressive, computational semantics. In my talk, I will present my on-going work on development of a generalized constraint-based grammar formalism for natural language processing. The approach integrates new theoretical developments of type theory within constraint-based lexicalized syntax-semantics interface.

1 Brief Intro to the Language and Type Theory of Acyclic Recursion

Moschovakis (2006) developed the language of acyclic recursion L^{λ}_{ar} (a proper extension of Montague's IL) to be used for modeling the logical concepts of meaning and synonymy, from the joint perspective of theory of computability and more adequate modeling of central semantic concepts. The semantics of L^{λ}_{ar} terms consists of two major layers: denotations and referential intensions, where the referential intensions are algorithms by which the denotations are computed.

The language of L^{λ}_{ar} undertakes computational ideas from semantics of programing languages, via an extended Higher Order Logic system, where algorithms are abstract mathematical objects defined by functions. L^{λ}_{ar} represents the algorithms for computing the denotations of meaningful expressions, at the levels of its semantics, formal syntax, and calculi. At the level of its own object language, algorithms are represented by terms constructed from a recursion operator where that "applies" over a *head* term A_0 and a set of acyclic assignments, called *body*, $\{p_1 := A_1, \ldots, p_n := A_n\}$, i.e., where-terms: $(A_0\ \mathsf{where}\{p_1 := A_1, \ldots, p_n := A_n\})$. The where-terms represent recursive computations that close-off, and are essential for encoding two-fold semantic information:

Denotational Semantics For any given semantic structure $\mathfrak{A}$, L^{λ}_{ar} provides (at most one) well-defined denotation function over terms and variable assignments. Thus, for every variable assignment g, every L^{λ}_{ar} term A *denotes* an object $\mathsf{den}(A)(g)$ in the domain frame of $\mathfrak{A}$, by respecting the type system. The promotion of states (for possible worlds, times, described situations, context information, etc.), in the syntax and semantics of L^{λ}_{ar}, enhances its expressive adequateness and facilitates computations of state dependent semantic representations. In particular, by including states in the types, expressions and semantic entities,

the L^{λ}_{ar} notion of denotation covers the classical distinction of *Montague extensionality* vs. *Montague intensionality* as a distinction of *locality* vs. *modality* in L^{λ}_{ar}.

Intensional Semantics The notion of intension in L^{λ}_{ar} coveres the computational aspect of the concept of meaning: the *referential intension* $\mathsf{Int}(A)$ of an expression A is the *algorithm* for computing its denotation $\mathsf{den}(A)$. Intuitively, an well-formed, meaningful expression has sense by carrying information about "how to compute" its denotation, i.e., expressions have sense by having been associated with instructions for acquiring what they denote in any given semantic structure. Two meaningful expressions are referentially (algorithmically) synonymous if their referential intensions are naturally isomorphic, i.e., their intensions are the same algorithms.

2 Summary of On-going Work

Type-theory of Recursion A class of specialized type-theoretic languages L^{λ}_{r}, with full recursion, will be defined for the purpose of: (1) logic foundations of CBG, and (2) semantic representations in CBG. A reduction calculus of L^{λ}_{r} will be developed by using the technical tools of the formal languages of recursion (Moschovakis, 1994–2006) and Situation Theory. The theory and calculi of denotational and intensional semantics of L^{λ}_{r} will be developed. The on-going work includes formalization of more fine granularity of the semantic concepts of denotations and referential intensions (i.e., algorithmic intensions) by type theory that represents:

1. Internal variation of states (situations) that occur inside terms of the formal language L^{λ}_{r} and take part in the corresponding denotations that are state dependent (e.g., terms with varying *resource situations*, as used in the terms of Situation Theory)
2. Underspecification of meaning
3. Partiality of meaning, i.e., of denotations and intensions. Such development requires using a version of the full language of recursion with partial functions and partial variable assignments.
4. Relational terms and relational semantic structures
5. The semantics of attitudes (such as *know*, *believe*, etc.), which is, currently, an open problem in L^{λ}_{ar}.
6. Implementations

3 Preliminary Results

The languages of recursion, e.g., L^{λ}_{ar}, with their calculi and theories, offer richly expressive systems for modeling various semantic notions and concepts in an inherently algorithmic way. In particular, various L^{λ}_{ar} terms represent different levels of underspecification and ambiguities in natural languages, for example, quantifier scopes. Some results in this direction have been investigated in Loukanova (2007). An example for integration of semantic representations within CBG by using L^{λ}_{r} has been offered in Loukanova (2008).

Some References

Loukanova, R. 2007. *Typed Lambda Language of Acyclic Recursion and Scope Underspecification.* In: Reinhard Muskens (Ed.) Workshop NDTTG Dublin, Ireland. Procs. p.73-89.

Loukanova, R. 2008. *Constraint Based Grammar and Semantic Representation with The Language of Acyclic Recursion.* In: Bel-Enguix. G. and M.D. Jimnez-Lpez (eds.). *Linguistics and Formal Languages.* Procs. ForLing 2008. pp.57-70,

Moschovakis, Yiannis N. 2006. *A logical calculus of meaning and synonymy.* Linguistics and Philosophy, v. 29. 2006. pp. 27 – 89.