Argument Structure and Referent Systems

Marcus Kracht
Bielefeld University
marcus.kracht@uni-bielefeld.de

Yousuf Aboamer
Bielefeld University
yousuf.aboamer@uni-bielefeld.de

1 Introduction

We describe here a theory of semantic composition, which integrates referent systems into argument structure. The main idea is that argument structure is the key component in linguistic signs, and that it provides the interface between syntax and semantics. To make this idea work for language, referent systems as defined in Vermeulen (1995) have been used. Referent systems (RS) are based on the idea that the identity of variables *across different representations* is irrelevant (see also Fine (2007)). Thus, whether or not two variables must be taken to be coreferential in the formation of a syntactic constituent must be effected by encoded linguistic cues, be they word order or a particular word form. This basic insight started the present investigation. Many adaptations had to be made in order to make the basic idea compatible with the requirements of natural language.

In RSs the natural merge operation in semantics makes all variables occurring free in the respective representations disjoint. Thus, in contrast to an idea promoted in Zeevat (1989), where variables shared across representations are taken to be pointing to the same object, here this is treated as an unwarranted assumption. In natural languages, however, there are clearly defined circumstances in which certain variables need to be identified during the merge operation, for example when a head is merged with its complement. The way this is done in RSs is that this fact is *explicitly* encoded by means of a shared name. This name is taken from a specified set of names.

This idea has been the starting point of the current theory. This theory takes it that argument structures are lists of argument identification statements (AISs), that need to be discharged one by one in order to reach the phrase level. A discharge is obtained by matching a single variable from the argument structure in both the functor and its argument. This match is subject to several conditions: (i) the variables occur in particular AISs that are visible during match, (ii) the variables have matching names, and (iii) no morphological conditions are violated.

In the present paper we shall simplify the actual theory in order to concentrate on its main engine, the argument structure. In particular, we shall not say much about the morphology. The theory is fully implemented, see the section on implementation for source code.

2 Semantic Structures

Semantic units are called signs. A *sign* is a triple $\langle m, \alpha, \Delta \rangle$, where m is a morpheme (= a set of morphs), α an argument structure and Δ a DRS. Both argument structures and morphs have a combinatorics, that is to say, they specify some number of arguments and modes how they will combine with them. Each morph in m has as many arguments as α. The sign combines any number of morphs that have the same semantics into a single morpheme. This accounts in particular for allomorphy, but may also be used for portmanteau realisation.

A *morph* is a pair $\mathfrak{m} = (\mathfrak{g}, \mathcal{A})$, where $\mathfrak{g}$ is its exponent, typically a string, and $\mathcal{A}$ a vector of so-called selectors (the length of which is called the dimension of $\mathfrak{m}$). The selectors state the number and morphological kind of arguments that the morph requires and in which way their exponents are to be combined with the exponent of the head. If exponents are strings, then we can state (i) that the argument is required to follow the head (the head is looking right for its argument), or (ii) that the argument is required to precede the head (head is looking left). Such statements are given for each argument independently; thus, it may be stated that the subject is to the left, while the direct object is to the right.

The name space is given as follows. We take a finite set A of *attributes*, and a finite set V of *values*, together with a function $o : A \to \wp(V)$. $o(n)$ is the set of *admissible values* for n (see Gazdar et al. (1988)). A legal *attribute value structure* (AVS) is a set of pairs (n, s), where $n \in A$ and $s \subseteq o(n)$. The sets represent underspecification. An absent name can be added in the form $(n, o(n))$. Thus we may see a legal AVS as a function $f : A \to \mathcal{P}(V)$ such that for all $n \in A$, $f(n) \subseteq o(n)$. A particular role is played by pairs $(n, \varnothing)$, which we write $(n, \star)$. $\star$ represents the *absence* of a value. A pair (n, s) *matches* a pair (n', s') iff (a) $n = n'$; (b) $s \cap s' \neq \varnothing$ (in which case $s \neq \varnothing$ and $s' \neq \varnothing$) *or* $s = s' = \varnothing$. Two legal AVSs f and g are *unifiable* if for all $n \in A$ $f(n)$ and $g(n)$ match. In that case, the unification is $(f \cap g)$, the pointwise intersection. A *name* is a fully specified AVS, that is, $f(n)$ is either a singleton or $\star$ for any given n.

In the sequel we use *functor* in place of *head*. When a functor F combines with an argument A, we require that there always be a semantic object, a variable, that needs to be shared between them. (This is called the *main variable*.) The mechanism will be spelled out below. Both the functor and the argument contain an argument structure. Let α be the AVS of the functor and β be the argument structure of the argument. The merge is denoted by $\alpha \bullet \beta$. An *argument structure* is a sequence of argument identification statements (AISs). An *AIS* takes the form

$$(x : \delta : \Sigma :: P)$$

where

1. x is a variable;
2. δ is a diacritic;
3. Σ is a pair of AVSs;
4. P is a pair of parameter AVSs.

We first spell out merge in case α and β contain single AISs: $\alpha = \langle \mu \rangle$, $\beta = \langle \nu \rangle$, where

$$\mu = (x : \delta : \Sigma :: P), \qquad \nu = (y : \epsilon : \Xi :: Q).$$

The diacritic δ contains specific instructions on merge. They specify among other whether x is supplied by the functor or not (we speak of the variable being *exported* or not, indicated by the presence or absence of the diacritic $\triangle$) and whether it is to be supplied by the argument or not (we speak of a variable being *imported* or not, indicated by the presence or absence of the diacritic $\triangledown$). This is a fourfold alternative; yet, since x is in the functor it must be imported by the functor. This reduces the space to two choices: if the functor does not export the variable, the AIS μ will be removed after merge. If it does, the merge $\mu \bullet \nu$ will replace it. This reproduces the argument/adjunct distinction (though it is more general since import names may be changed).

The argument variable y is likewise accompanied by a diacritic, ϵ. As the argument must export its variable, we are left with two options: ν is either importing y or not. In the latter case, the argument structure is not saturated, a case to which we shall return below.

Σ is a pair of AVSs. The first of these determines under which name the variable is imported, the second under which name the variable is exported. If the diacritic says "no export" then the second AVS is left empty (and is omitted in the notation); likewise when the diacritic says "no import" the first AVS

is left empty (or is simply omitted). Abstractly speaking, when a variable is both imported and exported, the functor specifies a function from import names to (sets of) export names. However, this function is restricted. The export name is actually underspecified, and so is the import name. Thus, the function is actually undefined or returns the same set of export names. However, the export AVS may contain a special value ✓ for an attribute, in which case the function returns the same value, ie is the identity. Thus the pair

$$[(\mathit{case}, \{\mathit{nom}, \mathit{acc}\}], [(\mathit{case}, ✓)]$$

returns $[(\mathit{case}, \{\mathit{nom}\})]$ given $[(\mathit{case}, \{\mathit{nom}\})]$ and $[(\mathit{case}, \{\mathit{acc}\})]$ given $[(\mathit{case}, \{\mathit{acc}\})]$. However,

$$[(\mathit{case}, \{\mathit{nom}, \mathit{acc}\}], [(\mathit{case}, \{\mathit{nom}, \mathit{acc}\})]$$

returns $[(\mathit{case}, \{\mathit{nom}, \mathit{acc}\})]$ for both inputs. This situation is quite frequent. For example, if an adjective A composes with a noun N that can either be nominative or accusative, then if the noun is accusative, so is the complex $[A\ N]$, and if the noun is nominative, so is the complex $[A\ N]$. Thus, the indeterminacy is only in the noun, and the indeterminate value is being passed up "as is", without adding indeterminacies for the new constituent.

The merge is simply function composition. For AVSs, this is spelled out as follows.

$$\Sigma = (f_1, g_1), \qquad \Xi = (f_2, g_2).$$

Primarily, f_1 and g_2 must be unifiable. If so, a new pair of AVSs is being computed. The details are straightforward, but tedious.

With the combination of Σ and Ξ defined, we turn to P and Q. Likewise, they consist of two parts, which are now parameter AVSs. These are sets of statements of the form (π, x), where π is some name (taken from a given set of parameter names), and x a variable. If $P = (U, V)$, and $Q = (U', V')$, then parameters will be unified during merge iff there is a name π such that U contains (π, x) and V' contains (π, y) (where y is potentially a different variable).

When all components match, we first get a pair of substitutions σ_1 and σ_2. The first substitution is to be executed on the first sign (the functor) and the second substitution is to be executed on the second sign (the argument). This substitution is effected on the semantics as well as the argument structure (main variable and parameter AVSs alike). After that a new sign is computed by:

1. applying each morph of the functor to each morph of the argument and keeping only the successful combinations;
2. computing the merge of the argument structures; and
3. taking the union (= Zeevat-merge) of the new DRSs.

We emphasise that the substitutions will unify variables on the basis of the AVSs alone. Thus, if the functor contains x as main variable and the argument contains $y \neq x$, and the merge succeeds, we will have $\sigma_1(x) = \sigma_2(y)$. However, if they do not get identified, then $\sigma_1(x) \neq \sigma_2(y)$. Likewise, if the first sign contains x and so does not the second, then $\sigma_1(x) \neq \sigma_2(x)$ *unless they are identified under merge*. This is a desired outcome.

Crucially, the number of arguments that are needed in morphology is the same as the number of arguments given by the argument structure. This says nothing else but that each time a syntactic nexus is established, some semantic merge must be performed—and conversely.

Here is a simple case of how this works. A verb may contain a specification of its subject in the following form.

$$/\texttt{rennen}/ : (x : \nabla : \begin{bmatrix} \text{pers}: \{1,3\} \\ \text{num}: \quad \{\mathit{pl}\} \\ \text{case}: \{\mathit{nom}\} \end{bmatrix} ::)$$

This says that the variable x is imported under merge under the name nominative-1st-plural or nominative-3rd-plural (a third option, that this is the infinitive, has been omitted). A pronoun in turn can contain the following

$$/\texttt{wir}/ : (y :\triangle: \begin{bmatrix} \text{pers}: & \{1\} \\ \text{num}: & \{pl\} \\ \text{case}: & \{nom\} \end{bmatrix} ::)$$

In that case the AISs merge successfully, and the complex has an empty argument structure. Moreover, the substitutions will be arranged such that $\sigma_1(x) = \sigma_2(y)$. Thus, the constituent /`wir rennen`/ is well formed, since they have the argument structures displayed above. By contrast, the following is not well-formed: /`uns rennen`/, even if the pronoun is given the following argument structure:

$$/\texttt{uns}/ : (x :\triangle: \begin{bmatrix} \text{pers}: & \{1\} \\ \text{num}: & \{pl\} \\ \text{case}: & \{acc\} \end{bmatrix} ::)$$

This is because accusative case blocks identification under merge, irrespective of the actual variable chosen.

3 Merge and Fusion

So far we have only talked about the case of a single AIS in each argument structure. Now we need to go full scale. First, say that an argument structure is *saturated* if none of its AISs imports a variable. There are now two kinds of merge: *proper merge* and *fusion*. Fusion is marked option. In a proper merge the argument structure of the complement is saturated. In a fusion it is not. The diacritic (in addition to the above fourfold choice) states whether the particular AIS allows for fusion.

Second, it is theoretically possible to identify several variables under merge. A merge is called n-ary if exactly n AISs are merged at the same time. A *monadic merge* is a 1-ary merge. A *polyadic merge* is an n-ary merge with $n > 1$. This option is used on control constructions. In the sentence /`Bert persuaded Mary to leave.`/ it is Mary who does the leaving, while in /`Bert promised Mary to leave.`/ it is Bert. This is accomplished by assigning the infinitive /`to leave`/ its semantic arguments, including the actor. However, the actor is not imported, it is exported on a par with the the event variable. Thus, when the higher verb identifies the event variable, it will also identify the actor. In this case, it can choose to either make it the same as the subject (in the case of /`promise`/) or the same as the theme (in the case of /`persuade`/).

Third, when the argument structures contain several AISs, access conditions apply. Suppose that

$$\alpha = (\mu_1, \mu_2, \cdots, \mu_k)$$

and that

$$\beta = (\nu_1, \nu_2, \cdots, \nu_\ell).$$

Then in a monadic merge

1. ν_1 must export a variable; and
2. for the largest (!) j such that $\mu_j \bullet \nu_1$ succeeds the following holds:
   - the global option of E-access is valid and $j = k$; or
   - the global option of G-access is valid and for no $i > j$, the variable of ν_i is a barrier.

In a polyadic merge, we inspect ν_2 for the next exported variable, and try to see if some μ_i, $i < j$, imports it. And so on.

To explain this, we need to resort to diacritics again. In a diacritic, a further specification is being made: whether the AISs can be skipped in merge or not (in the latter case it is a *barrier*). Now, a grammar is *E-access* if all AISs are barriers. A grammar is otherwise *G-access*.

To explain this notion, consider a language with case markers. Say a verb has an argument structure of the following form (think a verb in a language with subject agreement):

$$/\texttt{zumuten}/ : (x : \nabla : \left[\begin{array}{l}\text{pers}: \{1,3\} \\ \text{num}: \{pl\} \\ \text{case}: \{nom\}\end{array}\right] ::), (y : \nabla : [\ \text{case}:\{acc\}\ ] ::), (z : \nabla : [\ \text{case}:\{dat\}\ ] ::)$$

An NP that is marked for dative case can merge in both access conditions. For the last of the AISs matches the case of the NP. An NP with accusative can match only if it is allowed to skip the dative. An NP with nominative case can match only if it is allowed to skip both the dative and the accusative. So, given that arguments are to the left, under E-access only the following is grammatical:

NP-nom NP-acc NP-dat V

Given G-access and no variable is a barrier, we get 6 possibilities:

NP-nom NP-acc NP-dat V
NP-nom NP-dat NP-acc V
NP-acc NP-nom NP-dat V
NP-acc NP-dat NP-nom V
NP-dat NP-nom NP-acc V
NP-dat NP-acc NP-nom V

If word order is different, then matters can become more involved. For example, if only the nominative NP is to the left, there are only two patterns left:

NP-nom V NP-acc NP-dat
NP-nom V NP-dat NP-acc

In this way one can account for different word order patterns in languages. In Latin all word orders are well formed (though some may be dispreferred), while in English only one is permitted (/`John gave a book to Mary`/). German patterns like Latin, however verb-second complicates matters substantially.

4 Agreement

Grammar formalisms can be largely divided into those that base themselves on agreement (HPSG) and those that base themselves on cancellation (Categorial Grammar). The present framework combines cancellation with agreement. It performs *cancellation under agreement*. Thus, it is similar to some variant of UCG (unification categorial grammar, Calder et al. (1988)), however its categorial apparatus is greatly reduced.

It takes its motivation from the fact that agreement is one of the main inputs in deciding the association between variables across constituents. In many languages this is case; however, head marking regimes also exist. For example, in German a verb displays subject agreement. This fact not only regiments the proper form of the subject, as in the case of */`uns rennen`/ above. It also helps to disambiguate sentences as in /`Uns haben die Fragen geholfen.`/ "The questions were helpful for us.". The fact that the pronoun in first position, /`uns`/, is in the accusative, means that it cannot be subject. German allows for G-type access, so accusative NP may be sentence initial. On the other hand, /`die Fragen`/ ("the questions") may be both nominative and accusative. It turns out that only when

we decide them to be nominative is the sentence grammatical. This is borne out, assuming the following. (1) The verb /helfen/ has the argument structure

$$/\texttt{helfen}/ : (x : \nabla : \left[\begin{array}{l}\text{pers}: \{1,3\}\\ \text{num}: \ \{pl\}\\ \text{case}:\{nom\}\end{array}\right] ::), (y : \nabla : \left[\ \text{case}:\{acc\}\ \right] ::)$$

(2) The NP /die Fragen/ has two separate argument structures:

$$/\texttt{die Fragen}/ : (x :\triangle: \left[\begin{array}{l}\text{pers}: \ \{3\}\\ \text{num}: \ \{pl\}\\ \text{case}:\{acc\}\end{array}\right] ::)$$

and

$$/\texttt{die Fragen}/ : (x :\triangle: \left[\begin{array}{l}\text{pers}: \ \{3\}\\ \text{num}: \ \{pl\}\\ \text{case}:\{nom\}\end{array}\right] ::)$$

(3) The pronoun /uns/ has the structure

$$/\texttt{uns}/ : (x :\triangle: \left[\begin{array}{l}\text{pers}: \ \{1\}\\ \text{num}: \ \{pl\}\\ \text{case}:\{acc\}\end{array}\right] ::)$$

(We do not display the morphology here, for it is quite complicated.) Choosing a single entry in (2) with underspecified case gives the wrong result.

The calculus admits as many marking patterns for a verb as there are arguments that it takes. Therefore, it is possible to have no agreement (Chinese), subject agreement (German), subject and object agreement (Mordvin), and more. Similarly, adjectives may exhibit agreement with the noun they modify.

5 Parameters

The parameters are another important feature. Recall that an AIS contains a pair (U, V) of parameter AVSs. These consist of pairs (π, x), where x is a variable and π a parameter name. There is no prohibition for a variable to occur both as a parameter and as a main variable of the AIS. Also, variables may be assigned different parameter rules in import and in export AVSs.

A particularly instructive use of parameters is the phenomenon of sequence-of-tense (Abusch (1997), Ogihara (1996)). Recall that languages differ in how they use morphological tenses in subordinate clauses. Some languages require subordinate clauses to use nonpresent when the superordinate clause is in the past: /John said that he was ill./. Others require present tense if the subordinate clause is cotemporaneous with the event of the main clause. Russian is such a language. The phenomenon is explained as follows. Choose two parameter names, ρ ("reference time") and ε ("event time"). Put the argument structure of /said/ as

$$(e :\triangle: [\quad] :: \left[\begin{array}{l}\rho{:}\ t\\ \varepsilon{:}t'\end{array}\right]), (x : \nabla : [\quad] ::), (e' : \nabla : [\quad] :: [\quad], [\ \rho{:}t'\ ])$$

(irrelevant detail omitted) and in the semantics the statement $t' < t$ is added as well. Now, present tense may either be deictic (in which case it states that the event takes place at reference time) or it may be relational, in which case the event takes place at the event time of the higher predicate. To implement this, we give it the following argument structure:

$$(e :\triangle: [\quad] :: \left[\begin{array}{l}\rho{:}\ t\\ \varepsilon{:}t'\end{array}\right])$$

with $t = t'$. The idea is this. A verb of saying has a complement, and this complement needs to be situated in time. The higher verb decides whether the reference time of the embedded clause is its own reference time, or whether it is taken to be its event time. The first choice is realised in English. Reference time is not shifted. The second choice is realised in Russian. Reference time is shifted, because it is set to the event time of the higher clause.

We may combine this with a more fully fledged account of tense and aspect (as of Klein (1994)), but the core insight is as explained above, only that one more parameter is added and the choice of identification is multiplied further.

Time points are by far not the only parameters. From the evidence provided in Schlenker (2003) also worlds and speaker roles are shiftable, and therefore assume parameter roles. There are more: properties (to account for adjectives), locations, and even experiencers, eg for expressions of taste, see Kneer et al. (2017).

6 Implementation

Absent copying, the formalism leads to what is known as Multicomponent CFGs (Seki et al. (1991)). If copying is allowed, depending on further restrictions, the full power of Literal Movement Grammars can be achieved, equivalent to PTIME parseability (Groenink (1997), Kracht (2003)). Unlike Mel'cuk (2000) it is completely surface oriented.

The calculus has been faithfully implemented (in OCaml) and can be downloaded at

wwwhomes.uni-bielefeld.de/mkracht/referent/

This site contains the complete description together with a user manual as well as the source code and dictionaries for Hungarian nouns and basic Latin morphology.

Data is stored in a special XML-format. It handles UTF-8. Output can be rendered into XML or LaTeX and then shown via standard DVI or PDF viewers. This ensures high quality output. Based on a dictionary, one can either merge entries, or parse strings. The parser is a chart parser adapted for discontinuity. Morphs may be empty, but a special rank function ensures that this does not lead to infinite loops. The system has not primarily been designed for speed but to faithfully compute according to the theory. A chart parser has been chosen to determine all possible parses.

As an example we show here the entry of the verbal root /tang/ "to touch" of the Latin dictionary (as given in the online resource).

Id:tang

$\ast.(\circ,0)./\texttt{tac}/\langle[\ \text{BASE}:p;\text{LEVEL}:r;\text{STEM}:c\ ]:0\ {}^{0}\!\circ\rangle$

$\ast.(\circ,0)./\texttt{tang}/\langle[\ \text{BASE}:a;\text{LEVEL}:r;\text{STEM}:c\ ]:0\ {}^{0}\!\circ\rangle$

$\ast.(\circ,0)./\texttt{tetig}/\langle[\ \text{BASE}:f;\text{STEM}:c\ ]:0\ {}^{0}\!\circ\rangle$

$$\langle \text{e0} :\triangle: \begin{bmatrix} \text{ASP} & : & * \\ \text{CAT} & : & v \\ \text{MOOD} & : & * \\ \text{TENSE} & : & * \\ \text{TRS} & : & 2 \\ \text{VOICE} & : & * \end{bmatrix} :: \begin{bmatrix} \text{gf2} & : & \text{x1} \\ \text{gf1} & : & \text{x0} \end{bmatrix} \rangle$$

e0; x0; x1

$\text{touch}'(\text{e0}); \quad \text{agt}'(\text{e0}) = \text{x0};$
$\text{thm}'(\text{e0}) = \text{x1}.$

Parse terms:

$\ast$.tang : mr104

$\ast$.tang : mr105

$\ast$.tang : mr106

The first line is the identifier, an arbitrarily chosen string. The next lines specify the morphs. There are in total three morphs, corresponding to the three stems: /tang/, /tetig/, and /tact/. Each of them is accompanied by some morphological properties. Colours are used to highlight identical arguments across morphs and the argument structure.

Below that are found three boxes. The first contains the argument structure, the second the semantics (a DRS with head section and body), and finally a box containing the parse terms for the morphs. Three are listed, coindexed (by the stars of different shape) with the morphs given upstairs. Notice that the parse terms record the identifier as well as the morph contained in the entry.

The argument structure of a root is minimal: it exports an event variable, e. The category is specified as v (verbal) and the transitivity as 2. All other attributes have no value. The parameter AVS lists two parameters (for agent and theme).

7 Conclusion

The design criterion of this calculus has been to derive the complexity of phenomena from an interaction of several components, each of which are quite simple. In the morphology, we allow for discontinuous constituents, and each part can specify a limited context condition. Morphs are composed by piecing together the discontinuous parts in a specified manner. Allomorphy is explained by the fact that morphs specify context conditions; different morphs under the same morpheme may thus compete for the contexts.

The semantic algorithm is independent of this, however. It only takes into account whether the morphemes can be combined or not. Thus, in contrast to Morrill (2017), the discontinuity is not reflected in the categorical system. The main driver of semantic composition, by contrast, is the argument structure, which combines morphosyntactically supplied specification (AVSs) with semantic names of variables.

The third component, the semantics proper, is yet again independent. It is subjected to substitution before merge, where the substitution is calculated from the merge of argument structures.

The presented calculus is quite powerful. It has been designed to deal with the semantics in combination both with morphology and syntax. We do not draw a line between purely syntactic or purely morphological formation. Thus, there is no commitment to the lexical integrity principle. For example, it can be shown that case marking in Hungarian is best viewed as phrasal affixation. We have omitted the details of morphological analysis here. It is possible to decompose full paradigms (eg the nominal paradigm in Hungarian, the verbal paradigm in Latin) and give them the correct semantics, calculated bottom up.

The link with overt morphology and syntactic features is theoretically arbitrary. This is a drawback, which the calculus shares however with the overwhelming majority of frameworks. Further work is needed. We would ideally like to propose eg that in a language without overt gender marking (like Finnish or Hungarian) no attribute for gender exists. However, at this point, it is not prohibited to posit one, and not even mandatory that it have only one value.

References

Abusch, D. (1997). Sequence of tense and temporal *de re*. *Linguistics and Philosophy 20*, 1–50.

Calder, J., E. Klein, and H. Zeevat (1988). Unification Categorial Grammar: A Concise Extendable Grammar for Natural Language Processing. In *Proceedings of COLING 12, Budapest*, pp. 83–86.

Fine, K. (2007). *Semantic relationism*. London: Blackwell.

Gazdar, G., G. Pullum, R. Carpenter, T. Hukari, and R. Levine (1988). Category structures. *Computational Linguistics 14*, 1–19.

Groenink, A. (1997). *Surface without Structure. Word Order and Tractability Issues in Natural Language Analysis*. Ph. D. thesis, University of Utrecht.

Klein, W. (1994). *Time in Language*. London: Routledge.

Kneer, M., A. Vicente, and D. Zeman (2017). Relativism about predicates of personal taste and perspectival plurality. *Linguistics and Philosophy 40*, 37–60.

Kracht, M. (2003). *Mathematics of Language*. Berlin: Mouton de Gruyter.

Mel'cuk, I. (1993 – 2000). *Cours de Morphologie Générale*, Volume 1 – 5. Les Presses de l'Université de Montréal.

Morrill, G. V. (2017). Grammar logicised: relativisation. *Linguistics and Philosophy 40*, 119–163.

Ogihara, T. (1996). *Tense, Attitudes and Scope*. Dordrecht: Kluwer.

Schlenker, P. (2003). A Plea for Monsters. *Linguistics and Philosophy 26*, 29–120.

Seki, H., T. Matsumura, M. Fujii, and T. Kasami (1991). On multiple context–free grammars. *Theoretical Computer Science 88*, 191–229.

Vermeulen, K. F. M. (1995). Merging without mystery or: Variables in dynamic semantics. *Journal of Philosophical Logic 24*, 405–450.

Zeevat, H. (1989). A compositional approach to Discourse Representation Theory. *Linguistics and Philosophy 12*, 95–131.