Complex event representation in a typed feature structure implementation of Role and Reference Grammar

Erika Bellingham
University at Buffalo
ebelling@buffalo.edu

Abstract

Role and Reference Grammar (RRG) (Foley and Van Valin, 1984; Van Valin and LaPolla, 1997; Van Valin, 2005) is typologically general, semantically-driven syntactic framework. A major focus of RRG is a fine-grained typology of form-to-meaning mapping, emphasizing the syntactic and semantic structure of complex event descriptions. I introduce a constraint-based typed feature structure variant of RRG which exploits RRG's semantically-motivated syntactic backbone: the Layered Structure of the Clause (LSC). Each layer in the LSC is represented as a subtype of *sign*, and the unique combination of syntactic and semantic properties possessed by each layer is captured by a set of constraints on the appropriate *sign* subtype. Sentences are incrementally built up from predicates (represented semantically as event frames) to propositions using constructions to pass and combine information through the different layers (and juncture types) in the LSC. An English fragment of this constraint-based RRG is implemented using the Attribute Logic Engine (Carpenter, 1992).

1 Introduction

Role and Reference Grammar (RRG) offers a typologically motivated constructional approach to grammar, in which semantics plays a very significant role in syntactic structure. RRG's semantically motivated structural backbone, the Layered Structure of the Clause (LSC), offers several key advantages for capturing fine-grained semantic properties of classes of linguistic structures, particularly for complex sentences with multiple verbal elements. Semantic modifiers (adjuncts), as well as grammatical operators (e.g. tense, aspect, modals, negation), each apply to a specific layer of the clause, with the layer of application predicting both the position of the operator/modifier relative to the predicate (the nucleus of the clause), and the semantic scope of the operator/modifier. Complex event descriptions involving multiple verbal predicates vary in terms of their degree of both syntactic integration and semantic cohesion. The syntactic and semantic structures are isomorphically related to such a degree that an important conceptual property of the overall event representation - whether the event is represented as a single conceptual event (versus multiple conceptual events) - can be predicted based on the syntactic packaging.

The major effort towards formalizing RRG (Kallmeyer et al., 2013; Kallmeyer and Osswald, 2017; Osswald and Kallmeyer, 2018) uses the building blocks provided by Tree Adjoining Grammar (TAG) (Joshi and Schabes, 1997), with modified operations for combining trees. This paper introduces an alternative formalization possibility: a constraint-based variant of RRG, in which the layers in the LSC are implemented as typed feature structures. The focus is on building the backbone of syntactic and semantic representations to produce complex event representations that exploit the form-to-meaning mapping properties captured by the LSC, rather than on broad coverage of all syntactic possibilities. Drawing from Sign-based Construction Grammar (Sag, 2012), constructions are represented as typed feature structures with syntactic and semantic components. Events and entities are represented in a typed hierarchy of semantic frames, while the layers in the LSC are represented as subtypes of *sign*. Sentences are incrementally built from predicates using constructions to pass information between layers in the LSC.

Unification-based typed feature structure grammars (Carpenter, 1992) are mathematically well understood, computationally implementable and therefore easily testable, and are compatible with con-

Proceedings of the IWCS 2019 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 28–36
Gothenburg, Sweden, May 24, 2019.

struction grammars. A fragment an English grammar using the proposed constraint-based RRG is implemented with the Attribute Logic Engine (ALE) (Carpenter and Penn, 2001), a Prolog implementation of the formalism in Carpenter (1992). The paper is structured as follows. Section 2 summarizes the relevant components of Role and Reference Grammar, focusing on RRG's Layered Structure of the Clause. Section 3 briefly discusses unification-based construction grammars, and Section 4 introduces this constraint-based variant of RRG. The ALE implementation is discussed in Section 5.

2 The Layered Structure of the Clause in RRG

Syntactic structure in RRG is built around abstract syntactic categories with semantic correlates: the layers in the Layered Structure of the Clause. The LSC is based on two apparently universal contrasts in the structure of natural languages: (1) predicating elements versus non-predicating elements, and (2) arguments versus non-arguments. Predicating elements (canonically verbs) are contained within **nuclei**, predicating elements plus their arguments are contained within **cores**, (most) non-arguments (adjuncts) are contained within the **core periphery**, and cores plus their peripheries are contained within the **clause**. Each layer is compatible with certain types of grammatical operators (Section 2.1) and semantic modifiers (Section 2.2): these operators or modifiers scope over a particular layer, which becomes especially relevant when considering the structure and interpretation of complex event descriptions (Section 2.3).

2.1 Grammatical operators

Grammatical categories (e.g. tense, aspect, modality, negation) are modelled in RRG as 'operators', each of which may apply only to a specific layer in the LSC. Not all languages have all operators: Standard German lacks grammatical aspect, English lacks grammatical evidentiality, Yucatec Maya lacks grammatical tense, and so on. Nuclear operators (e.g. aspect) modify something about the event/action/state without reference to any of the event participants, and therefore have scope only over the nucleus. Core operators (e.g. deontic modals) modify some property of the involvement of the participants in the event, and have scope over the core. Clausal operators, which modify propositions, fall into two distinct categories: tense and status operators situation the proposition expressed by the clause temporally and along the irrealis-realis dimension respectively; while illocutionary force and evidentiality are more sentential in nature, scoping over the first category of clausal operators and applying only to main clauses which are immediately dominated by a sentence node.

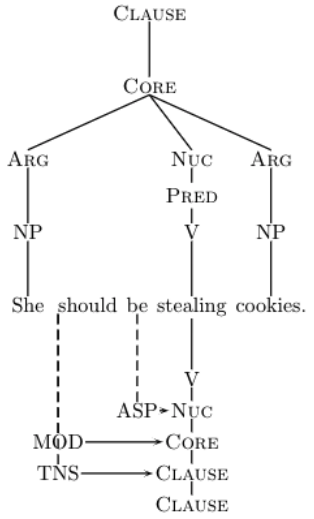


Figure 1: English clause structure with operator projection

An English clause with three grammatical operators is shown in Figure 1. The syntactic and the operator structures are projected above and below the sentence respectively. The aspectual operator expressed by the *be V-ing* construction applies to the nucleus. The deontic modal operator expressed by the *should* auxiliary applies to the core. The tense operator is expressed by the finite auxiliary *should*.

2.2 Semantic modifiers

The clause, core, and nucleus each have their own periphery, containing modifiers that combine with the non-peripheral constituent in that layer. Modifiers include prepositional phrases and adverbs. Just as the layer of application for each type of grammatical operator is semantically motivated, so too is the layer of application for each semantic category of modifier. Aspectual adverbs (e.g. *completely* or *continuously*) modify the nucleus and occur in the nuclear periphery; pace adverbs (e.g. *quickly*), manner adverbs (e.g. *carefully*) and temporal and locative modifiers (e.g. *yesterday*, *on Friday*, *in the kitchen*, *in an hour*) modify the core and occur in the core periphery, and epistemic and evidential adverbs (e.g. *evidently*, *probably*) modify the clause and occur in the clause periphery. The layering of modification also captures restrictions on the possible orderings for modifiers: if a modifier applies at a higher layer

than another modifier, it cannot occur between that lower modifier and the verb. For example in (1c), but not in in (1a-b), the core modifier ***on Friday*** intervenes between the nucleus *destroyed* and the nuclear modifier ***completely***, and so the sentence is anomalous.

(1) a. Jane *destroyed* the painting **completely**$_{NUCMOD}$ **on Friday**$_{COREMOD}$.
b. Jane **completely**$_{NUCMOD}$ *destroyed* the painting **on Friday**$_{COREMOD}$.
c. # Jane *destroyed* the painting **on Friday**$_{COREMOD}$ **completely**$_{NUCMOD}$.

The application of operators and modifiers is also relevant when considering the structure of complex sentences with multiple predicates. This is discussed further in Section 2.3.

2.3 The structure and interpretation of complex event descriptions

While simple sentences are built around a single predicating element, with a single nucleus, core, clause and sentence, more complex sentences can contain multiple predicating elements, requiring a more complex syntactic structure to combine them. Traditional theories of grammar typically assume that syntactic units are either joined together via **coordination** inside a larger unit, or via **subordination** (with one unit embedded in the other, as either an argument or a modifier). RRG recognizes a third possible nexus type, **cosubordination**, which shares some properties with coordination, and some with subordination.

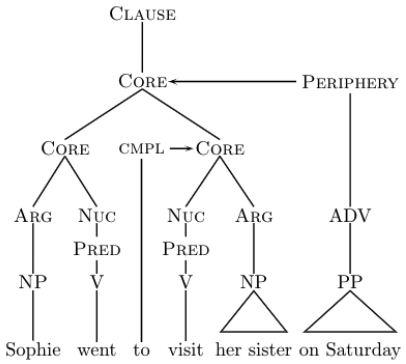


Figure 2: Core cosubordination

In cosubordination (Figure 2), two units of the same type are joined together to form a unit of that same type (i.e. two nuclei form a nucleus, two cores form a core, or two clauses form a clause). The operators and periphery (modifiers) that apply to that type are shared across both units (only the mother unit, and not the daughter units, has its own periphery and operators). In coordination, each unit has its own operators and periphery, and the two units join together to form a larger unit (e.g. two cores form a clause, two clauses form a sentence). A sentence instantiating core coordination is shown in Figure 3. In non-subordinate core junctures, each core has its own argument structure, but an argument can be shared between the two cores via a control or raising relationship (argument sharing is obligatory in core cosubordinations, and possible but not obligatory in core coordinations). In this case, the argument NP *Sophie* occurs in the first core, but is also an argument of the second core (the Actor in the visiting event).

The possible linkage relations (juncture-nexus types) can be arranged hierarchically in terms of the strength of the syntactic bond between the units: from closer to being integrated into a single unit (e.g. a complex predicate - nuclear cosubordination), to closer to being two separate units (e.g. sentential coordination). Nuclear junctures are the most compact, followed by core, clausal and sentential. Within each layer, cosubordination is the tightest nexus type, followed by subordination and then coordination. The hierarchy of syntactic relations aligns with a hierarchy of semantic relations according to the degree of semantic cohesion between the the two units: closer to being conceptualized as facets of a single event/action, or to being conceptualized as two distinct events/actions.

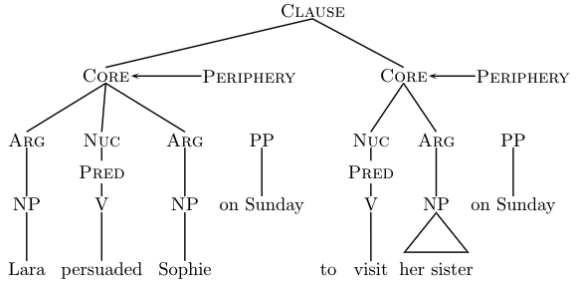


Figure 3: Core coordination

A cross-linguistically applicable breakpoint in the hierarchy of linkage relations was identified by Bohnemeyer and Van Valin (2017): constructions which combine two units using a core cosubordination linkage (or tighter) have the property of construing the described event as a single conceptual event, while constructions which combine two units using a core coordination linkage (or weaker) have the property of construing the described event as consisting of multiple conceptual events. The Macro Event Property

(MEP) (Bohnemeyer et al., 2007) is a possible semantic property of a syntactic construction (containing an event description). Constructions which have the MEP package an event representation such that any time-positional modifiers necessarily have scope over all of the subevents in the construction: the event is construed as more like a single event. Constructions which lack the MEP package an event representation such that multiple time-positional modifiers are possible (such that they each have scope over a different subpart of the complex event): the complex event is construed more like multiple events. (2) exemplifies this property using English infinitival constructions.

(2) a. Lara persuaded Sophie on Thursday to visit her sister on Saturday. (*Core coordination*)
b. Sophie went (#yesterday) to visit her sister today. (*Core cosubordination*)

Nuclear junctures necessarily have the MEP, as do cosubordinate core junctures, while core coordination, core subordination, and all clausal and sentential junctures lack the MEP. This fits neatly with the idea of a shared periphery: because time-positional modifiers occur in the core periphery, and core cosubordinations (as well as all nuclear junctures) share a single core periphery, time-positional modifiers necessarily have scope over the entire complex event. This distinction between complex events descriptions that have the MEP, construing the event as a single *macro event*, and those that lack the MEP, and construe the complex event as multiple *macro events*, is exploited in the semantic representation of the implemented grammar (see Section 4.3.1).

3 A typed feature structure approach to Construction Grammar

The term Construction Grammar describes a family of grammatical theories in which constructions (defined as form-meaning pairings) are basic units. Sign-based Construction Grammar (SBCG) (Sag, 2012) combines insights from HPSG (Pollard and Sag, 1994) and Berkeley Construction Grammar (Fillmore and Kay, 1996). The grammar signature is an inheritance hierarchy of types. Each type has an associated set of features, and the value of each feature is itself typed. Features and their type constraints are inherited in the type hierarchy: subtypes must have all the features and value type restrictions of their ancestors, plus any additional restrictions introduced for that subtype. Words and phrases are **signs** (pairings of form and meaning). Local trees are encoded as feature structures as shown in (3).

(3) $\begin{bmatrix} \textsc{mother} & np \\ \textsc{dtrs} & \langle det, n \rangle \end{bmatrix}$

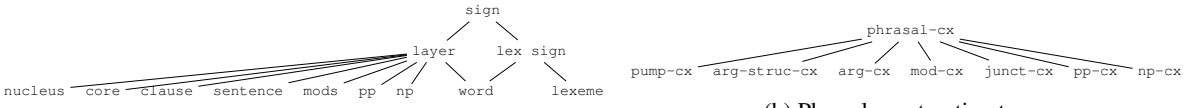
Although the proposed constraint-based RRG variant shares with SBCG the use of Frame Semantics (Fillmore, 1982), frames are embedded hierarchically rather than as a flat list. The focus of the present work is on the structure of complex event representations (distinguishing event frames, macroevents and propositions), rather than on the ability to handle phenomena such as quantifier scope ambiguity.[1]

4 A constraint-based variant of Role and Reference Grammar

Role and Reference Grammar (Van Valin, 2005) traditionally recognizes constructions as an important part of the grammar, but restricts them to cover only the more idiosyncratic structures in a language. Unlike constructions in SBCG and other unification-based construction grammars, RRG's constructions rely heavily on additional linking rules and template selection principles (Van Valin, 2005, p.244). RRG posits an inventory of abstract syntactic templates and linking relations (juncture-nexus types) which are distinct from grammatical construction types: these templates/relations may appear on the syntactic side of a constructional schema. Unlike standard RRG, where only idiosyncratic phenomena are captured with explicit constructional schemas, in the proposed constraint-based RRG variant all syntactic structures are captured as constructions (of varying degrees of abstraction).

[1] A flatter structure closer to Minimal Recursion Semantics (Copestake et al., 2005) may be considered in the future.

The various layers (nucleus, core, clause, sentence) are represented as subtypes of *layer*, a subtype of *sign* (Figure 4a). Other layer subtypes are *pp* (prepositional phrase) and *np* (noun phrase). As in SBCG, phrasal constructions (Figure 4b) are represented as feature structures with MOTHER and DTRS features.



(a) Sign types

(b) Phrasal construction types

Figure 4: Type hierarchies in a constraint-based variant of RRG

The type signature for *sign* is shown in (4): objects of type *sign* have a SYN feature of type *syn-obj*,[2] and a SEM feature of type *sem-obj*. The subtype *layer* additionally introduces the boolean feature PERIPHERY, to indicate whether any peripheral modification has already been applied to that layer (cosubordination constructions require that the daughter layers have not yet had modification applied).

(4) $\begin{bmatrix} \textit{sign} \\ \text{SYN} \quad \textit{syn-obj} \\ \text{SEM} \quad \textit{sem-obj} \end{bmatrix}$

(5) $\begin{bmatrix} \textit{layer (↑ sign)} \\ \text{PERIPHERY} \quad \textit{boolean} \end{bmatrix}$

(6) $\begin{bmatrix} \textit{pumping-cx (↑ phrasal-cx)} \\ \text{MOTHER} \quad \begin{bmatrix} \textit{layer} \\ \text{SYN} \quad \boxed{1} \end{bmatrix} \\ \text{DTRS} \quad \left\langle \begin{bmatrix} \textit{layer} \\ \text{SYN} \quad \boxed{1}\begin{bmatrix} \text{CLM} & \textit{clm-none} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$

Pumping constructions pass each non-complex layer up to the next layer. DTRS in a pumping construction is a singleton list containing one *layer* type (e.g. a *core*), and the MOTHER is the *layer* above (e.g. a *clause*). Syntactic (SYN) features are passed from the daughter to the mother, while some elements of the semantics (SEM) are changed from layer to layer (discussed in Section 4.3). The transition from nucleus to core is not captured in a pumping construction, but in an argument structure construction, as changes must be made to the valence list (SYN|VAL) (SYN values remain constant for all other layer transitions). Instead of RRG's standard linking algorithms, **argument structure constructions** assign the appropriate number and ordering of arguments positions to a core, and **argument constructions** add appropriate arguments to those positions.[3]

Figure 5 shows the derivation of a simple intransitive sentence (*Pat laughed*). A sentence is produced from a clause through a pumping construction (or via a more complex juncture). A minimum of four pumping constructions (or more complex juncture constructions) plus an argument construction are required to produce a sentence from a verb. As a consequence, there is a core node for every argument, unlike standard RRG. This tree structure captures more of the argument linking process than a standard RRG tree, where the linking algorithm is represented separately.

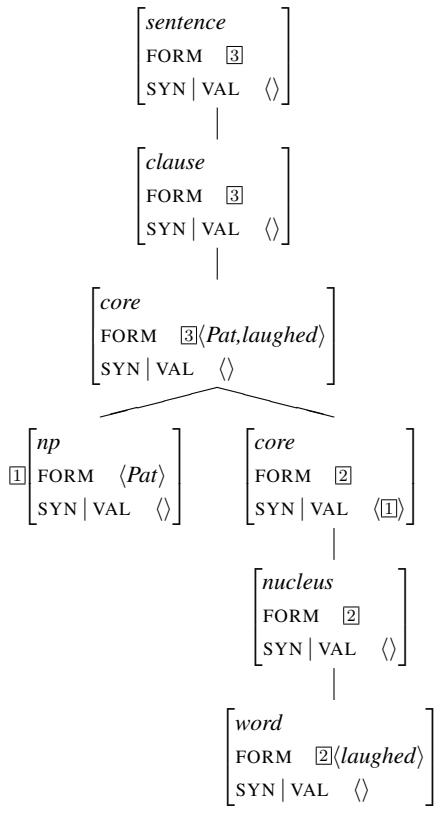


Figure 5: Simplified derivation *Pat laughed*

Juncture constructions combine two units of the same layer in coordinate or cosubordinate nexus. For example, a core cosubordination construction combines two daughter cores and produces a mother core, while a core coordination construction combines two daughter cores and produces a mother clause. Juncture constructions may place both syntactic (e.g. verb form, presence of a particular clause linkage

[2] I omit further discussion of the *syn-obj* type here for space reasons.

[3] This is English-centric, and may be adjusted in the future to allow for broader typological coverage.

marker) and semantic (e.g. subtypes of a certain semantic event frame) restrictions on their constituents.

4.1 Operators and modifiers

The *sem-obj* type (7) has a feature OPERATORS (OPS), which records the value of each of the nuclear, core and clausal operators that have been incorporated into a representation. These operators are then passed to the representation of the relevant semantic unit (see Section 4.3.1) at the time it is generated. In this way, the grammar is able to handle operators that occur on the surface between elements of a lower layer in the LSC (e.g. clausal operators inside the core, or tense morphemes attached to predicates).

Modifier constructions combine a layer in the LSC with something in its periphery. As discussed in Section 2.2, different semantic categories of verbal modifier apply to different layers in the clause. This restriction is captured by having a different modifier construction for each semantic category of modification, and restricting each modifier construction to apply to a particular layer (nucleus, core, clause or sentence). As modifiers combine a layer in the LSC with something in its periphery, all modifier constructions constrain the mother's PERIPHERY to $+$. This value is effectively reset each time a new layer is reached, as it is not passed from daughter to mother. While this approach does not allow for modifiers applicable to one layer to be interspersed between the elements of a lower layer, this could be improved in the future by adjusting the representation of modifiers to be more like the operator representation.

4.2 Comparison to a Tree Adjoining Grammar-based RRG formalization

The TAG-inspired approach to RRG formalization pursued by Osswald and Kallmeyer (2018) produces trees that are more consistent with standard RRG that the approach I have proposed. In the TAG approach, the trees are flatter, represent the fully derived structure rather than the steps involved in derivation, and capture the extended domain of locality in a way that is more consistent with standard RRG: long distance dependencies are captured using a 'wrapping substitution' mechanism, in which the predicate-argument structure that is the source of the long distance dependency is wrapped around the intervening elements. In the present approach, long distance dependencies would need to be percolated through the layered structure of the clause, as in Pollard and Sag (1994).

4.3 Semantics representation

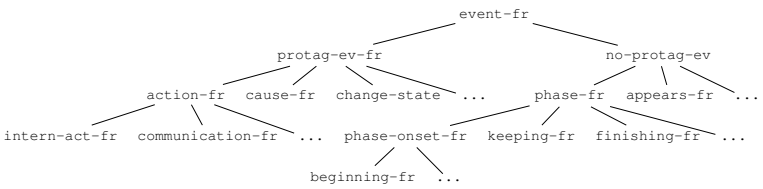


Figure 6: A section of the event frame type hierarchy

Although not explicitly arranged hierarchically, or using typed feature structures, the verbal representations in standard RRG can be straightforwardly adapted to hierarchically organized feature structure representations (*semantic frames*). RRG's lexical decomposition semantics already organizes verbs into a number of classes. A fragment of the proposed event frame hierarchy is shown in Figure 6. Each event frame is associated with a set of boolean aspectual features (STATIC, DYNAMIC, TELIC and PUNCTUAL), following Van Valin (2005). These features are used to restrict the application of constructions that are sensitive to the aspectual properties of their constituents, such as phase constructions (e.g. *begin to X*) and duration (e.g. *for an hour*) or time-to-completion modifiers (e.g. *in two hours*). Participant roles are already hierarchically organized in RRG, so that most roles are fairly specific but can also be categorized into more coarsely grained thematic relations and macro roles (see Van Valin, 2005, p.54). Verbs describing *activities* are represented with **do'** in the logical representation. This could be transformed into an abstract ACTOR role in an event frame, to be realized as EATER, RUNNER, TALKER and so on in specific event frames.

4.3.1 Semantic representation: entities, event frames, macroevents and propositions

Extending the idea of the Macro Event Property (Bohnemeyer et al., 2007; Bohnemeyer and Van Valin, 2017), a distinction is made between event frames, macroevents, and propositions. Macroevents are com-

posed from the event frame in the nucleus, any additional event frame contributed by the argument structure construction, and any core cosubordination junctures. As the macroevent is built up from the nucleus to the core to be passed on to the clause, it is represented in SEM|FRAMES|BUILD-MACROEVENT (abbreviated to SEM|FRMS|B-MACRO). Once a core becomes a clause, either through a pumping construction (10), or through a core coordination construction, the *macro-event-obj* in BUILD-MACROEVENT is transferred to the MACRO-EVENTS (M-EVS) list in the first *proposition-obj* in the PROPOSITIONS (PROPS) list. The type signatures for *sem-obj*, *macro-event-obj* and *proposition-obj* are shown in (7-9):

(7)
$$\begin{bmatrix} \textit{sem-obj} & \\ \text{INDEX} & \textit{index} \\ \text{FRMS} & \begin{bmatrix} \text{ENTITIES} & \textit{list(ref-descriptor-obj)} \\ \text{PROPS} & \textit{list(proposition-obj)} \\ \text{PROP-RELS} & \textit{list(proposition-rel-obj)} \end{bmatrix} \\ \text{OPS} & \begin{bmatrix} \text{NUCLEAR-OPS} & \textit{nuclear-ops-obj} \\ \text{CORE-OPS} & \textit{core-ops-obj} \\ \text{CLAUSAL-OPS} & \textit{clausal-ops-obj} \end{bmatrix} \end{bmatrix}$$

(8)
$$\begin{bmatrix} \textit{macro-event-obj} & \\ \text{EV-FRMS} & \textit{list(frame)} \\ \text{MODS} & \textit{list(frame)} \\ \text{OPERATORS} & \textit{core-ops-obj} \\ \text{ASPECT-PROF} & \textit{aspect-obj} \\ \text{INDEX} & \textit{index} \end{bmatrix}$$

(9)
$$\begin{bmatrix} \textit{proposition-obj} & \\ \text{M-EVS} & \textit{list(macro-ev-obj)} \\ \text{MODS} & \textit{list(frame)} \\ \text{OPERATORS} & \textit{clausal-ops-obj} \\ \text{M-EV-RELS} & \textit{list(macroevrel-obj)} \\ \text{INDEX} & \textit{index} \end{bmatrix}$$

$list(\tau)$ represents a list containing only elements of type τ. Propositions (9) are composed from the macroevent/s contributed by the core/s that feed into a sentence. Core coordination constructions produce two macroevents, while core cosubordination constructions produce only one: the events contributed by each daughter core are integrated into the semantics of a single macroevent in the juncture construction.

(10)
$$\begin{bmatrix} \textit{core-to-clause (}\uparrow \textit{pumping-cx)} & \\ \text{MTR} & \begin{bmatrix} \textit{clause} & \\ \text{SEM} \mid \text{FRMS} & \begin{bmatrix} \text{ENTITIES} & \boxed{1} \\ \text{M-EVS} & \boxed{5} \oplus \boxed{2} \\ \text{PROPS} & \boxed{4} \\ \text{PROP-RELS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \text{DTRS} & \left\langle \begin{bmatrix} \textit{core} & \\ \text{SEM} \mid \text{FRMS} & \begin{bmatrix} \text{ENTITIES} & \boxed{1} \\ \text{PROPS} & \boxed{4} \\ \text{PROP-RELS} & \boxed{3} \\ \text{BUILD-M-EV} & \boxed{2} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$$

(11)
$$\begin{bmatrix} \textit{core-cosub (}\uparrow \textit{juncture-cx)} & \\ \text{MTR} & \begin{bmatrix} \textit{core} & \\ \text{SEM} & \begin{bmatrix} \text{INDEX} & i \\ \text{OPERATORS} & \boxed{1} \\ \text{FRMS} \mid \text{B-MACRO} \mid \text{ASPECTP} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{DTRS} & \left\langle \begin{bmatrix} \textit{core} & \\ \text{PERIPHERY} & - \\ \text{SEM} & \begin{bmatrix} \text{INDEX} & i \\ \text{OPERATORS} & \boxed{1} \\ \text{FRMS} & \begin{bmatrix} \text{B-MACRO} \mid \text{ASPECTP} & \boxed{2} \\ \text{ENTITIES} & \langle\rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}, \begin{bmatrix} \textit{core} & \\ \text{PERIPHERY} & - \end{bmatrix} \right\rangle \end{bmatrix}$$

The precise relationship between event frames (contributed either by a predicate or by a construction) is specified in the constraints of each construction. Core layer modifiers are added to the MODS list of the *macro-event-obj*. The onset-phase construction shown in (12) specifies that the first core must contribute a *phase-onset-fr*, while the second core must contribute an *event-fr* which fills the EVENT role of the *phase-onset-fr*. The ultimate semantic output of a sentence is a list of entities, and a list of propositions (plus proposition relations, in the case of sentences containing multiple clause nodes).

(12)
$$\begin{bmatrix} \textit{ccosub-onset-phase (}\uparrow \textit{core-cosub)} & \\ \text{MTR} & \left[\text{SEM} \mid \text{FRMS} \mid \text{B-MACRO} \mid \text{EV-FS} \quad \left\langle \boxed{3} \begin{bmatrix} \textit{phase-onset-fr} \\ \text{EVENT } \boxed{4} \end{bmatrix} \right\rangle \right] \\ \text{DTRS} & \left\langle \left[\text{SEM} \mid \text{FRMS} \mid \text{B-MACRO} \mid \text{EV-FS} \ \langle \boxed{3} \rangle \right], \left[\text{SEM} \mid \text{FRMS} \mid \text{B-MACRO} \mid \text{EV-FS} \ \langle \boxed{4} \textit{ event-fr} \rangle \right] \right\rangle \end{bmatrix}$$

5 Implementation in ALE

The Attribute Logic Engine (ALE) is a freeware logic programming and grammar parsing and generation system written in Prolog. For a full description of ALE, see Carpenter and Penn (2001). The proposed constraint-based RRG-variant is implemented for a fragment of English.[4] The current fragment contains a restricted set of 419 lexical items and 45 leaf node constructions. A semi-automatic expansion of the lexicon using FrameNet is planned. The implementation includes: a type-hierarchy, containing all types and their features; a list of constraints on types; a lexicon; a list of lexical rules; a list of phrase rules which map constituents to the daughters of a compatible construction, thus generating a mother constituent; a list of declarative statements which constrain the parser so that only the leaf construction types in the hierarchy (and not the more abstract parent types) can be parsed; and macros to simplify repeated chunks of code. Listing 1 shows the implementation of the constraints on the abstract *core-cosubordination construction* (11). Listing 2 shows the output of the `recsem()` command, which parses a sentence and displays the semantic output of a sentence: a list of entities and a list of propositions.

Listing 1: Core cosubordination constraints in ALE

```
coreCosub cons
    (mother:(core,
        syn:(category: Cat,
            clmm: clm_none),
        sem:(index:I,
            operators: 0,
            frames:(buildMacroEvent:(aspectprofile:Asp)))),
    dtrs:[(core,
          periphery: minus,
          syn:(category: Cat,
              clmm:clm_none),
          sem:(index:I,
              operators: 0,
              frames:(buildMacroEvent:(aspectprofile:Asp),
                      entities:       [])),
            (core, periphery: minus)]).
```

Listing 2: Semantic output for *The cat began to purr.*

```
sem_output
ENTS    referentdescriptor
        BOUNDED plus
        CAT cat_fr
            ENTITY [0]
        DEF definite
        IX [0]
        NUMBER sg
PROPS   proposition
        MACROEVS    macroEvent
                    ASPECTPROF [1] achievement
                    EV-FRMS beginning_fr
                        ASPECT [1]
                        PHASE_EV purring_fr
                            PROTAG [0]
                        SIT [2]
        OPERATORS   clauseops
                    TENSE past
        PROP_INDEX [2]
```

Constructions are parsed via phrase rules. Phrase rules in ALE specify an output, one or more constituents (`cat>`) in the order they must be encountered, and goals (`goal>`) which evaluate Prolog predicates with respect to variables from the output or constituents of the phrase rule. In this implementation, the type of each constituent is always *phrasal-cx*, and `cat>` is only ever sensitive to the constituent's MOTHER value. The phrase rule output is also a construction, and each item in its DTRS list must unify with the MOTHER of a particular constituent. The possible output construction types are restricted via the `goal`: only construction types which satisfy a specified Prolog predicate (defined in a list of declarative statements) are eligible. In this way, parsing is restricted to only leaf node construction types, and not their more abstract/underspecified supertypes.

6 Future work

The current implementation of this constraint-based variant of RRG has focused primarily on: a) building out the backbone of the layered structure of the clause; and b) the structure and semantic representation of complex event descriptions. Two major additional features are currently being developed: the incorporation of information structure (the third projection in RRG's LSC model), which will extend the model beyond parsing declarative sentences with broad focus; and aspectual coercion (c.f. Michaelis, 2004), which will increase the flexibility of the grammar in parsing event descriptions that do not make use of the default aspectual profile of a particular event.

[4]The full code plus user manual are available on Github: https://github.com/ebellingham/constraint-based-rrg

References

Bohnemeyer, J., N. J. Enfield, J. Essegbey, I. Ibarretxe-Antunano, S. Kita, F. Lüpke, and F. K. Ameka (2007). Principles of event segmentation in language: The case of motion events. *Language 83*(3), 495–532.

Bohnemeyer, J. and R. D. Van Valin (2017). The macro-event property and the layered structure of the clause. *Studies in Language 41*(1), 142–197.

Carpenter, B. (1992). *The Logic of Typed Feature Structures*. New York, NY, USA: Cambridge University Press.

Carpenter, B. and G. Penn (2001). *ALE: The attribute logic engine user's guide [Version 3.2.1]*. Carnegie-Mellon University.

Copestake, A., D. Flickinger, C. Pollard, and I. A. Sag (2005). Minimal Recursion Semantics: An Introduction. *Research on Language and Computation 3*(2-3), 281–332.

Fillmore, C. J. (1982). Frame Semantics. In *Linguistics in the Morning Calm*, pp. 111–137. Hanshin Publishing Co.

Fillmore, C. J. and P. Kay (1996). Construction grammar coursebook.

Foley, W. and R. D. Van Valin (1984). *Functional syntax and universal grammar*. Cambridge: Cambridge University Press.

Joshi, A. K. and Y. Schabes (1997). Tree-adjoining grammars. In *Handbook of formal languages*, pp. 69–123. Springer.

Kallmeyer, L. and R. Osswald (2017). Combining predicate-argument structure and operator projection: Clause structure in role and reference grammar. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, pp. 61–70.

Kallmeyer, L., R. Osswald, and R. D. Van Valin (2013). Tree wrapping for role and reference grammar. In *Formal Grammar*, pp. 175–190. Springer.

Michaelis, L. A. (2004). Type shifting in construction grammar: An integrated approach to aspectual coercion. *Cognitive Linguistics 15*(1), 1–67.

Osswald, R. and L. Kallmeyer (2018). Towards a formalization of role and reference grammar. In R. Kailuweit, L. Knkel, and E. Staudinger (Eds.), *Applying and Expanding Role and Reference Grammar*, NIHIN Studies, pp. 355–378. Freiburg: Albert-Ludwigs-Universität, Universitätsbibliothek.

Pollard, C. and I. A. Sag (1994). *Head-driven phrase structure grammar*. University of Chicago Press.

Sag, I. A. (2012). Sign-based construction grammar: An informal synopsis. In *Sign-based construction grammar*, Volume 193, pp. 69–202. CSLI: CSLI Publications.

Van Valin, R. D. (2005). *Exploring the syntax-semantics interface*. Cambridge: Cambridge University Press.

Van Valin, R. D. and R. J. LaPolla (1997). *Syntax*. Cambridge: Cambridge University Press.