Modeling Clausal Complementation for a Grammar Engineering Resource

Olga Zamaraeva
University of Washington
olzama@uw.edu

Kristen Howell
University of Washington
kphowell@uw.edu

Emily M. Bender
University of Washington
ebender@uw.edu

Abstract

We present a grammar engineering library for modeling objectival declarative clausal complementation patterns attested cross-linguistically. Our primary contribution is positing a set of syntactico-semantic analyses couched within a variant of the HPSG syntactic formalism and integrating them with a variety of phenomena already implemented in a grammar engineering toolkit. We evaluate the addition to the system on testsuites from genetically diverse languages that were not considered during development.

1 Introduction

Grammar engineering is the modeling of language rules in a machine-readable fashion, such that the resulting system (the *grammar*) can parse grammatical strings while not *overgenerating* (licensing incorrect or spurious analyses). Linguistic grammar engineering prioritizes *precision* (how many parses are syntactically and semantically correct with respect to the linguistic formalism) over *recall* (how many input strings get parsed).

A *precision grammar* documents and models linguistic phenomena on the one hand, and is a program that parses and generates, on the other. This makes precision grammars a resource for rigorous linguistic hypothesis testing as well as for NLP tasks. While precision grammars are expensive—broad-coverage grammars like Flickinger 2000, 2011 or Siegel et al. 2016 take years to build—there are efforts to automate this process. We present a contribution to one such initiative, adding a clausal complements library to a grammar engineering starter toolkit.

Clausal complements are a kind of subordinate clause. Modeling subordinate clauses in general and clausal complements in particular is important not only because of their corpus frequency, but also because of their interaction with other phenomena. A grammar that cannot support subordinate clauses will fail to provide analyses for a big portion of a typical corpus and will not support the grammar engineer in exploring how well the analyses used for modeling simple clauses generalize.

To fill this gap, we incorporate a cross-linguistic account of clausal complements into a grammar engineering questionnaire and customization system. The questionnaire elicits typological information about hypothetically any language and the customization system outputs a starter precision grammar to specifications. In this way, the toolkit supports the rapid development of precision grammars, giving both novice and experienced grammar developers a means to create a grammar fragment customized to their language of interest and ready for extension to broader coverage.

In the meta-grammar engineering context, an *analysis* is a set of theoretically-grounded elements which comply with the requirements of the specific implementation framework and are used by the grammar engineering system so as to produce functioning grammars. In our case, the analysis is couched within Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag, 1994) and Minimal Recursion Semantics (MRS; Copestake et al., 2005) and is implemented as an extension to the LinGO Grammar Matrix (Bender et al., 2002, 2010). Our main challenges lie in accounting for a large space of typological possibilities while at the same time integrating our analysis of clausal complements into a complex system which outputs streamlined grammars. We start by motivating the choice of the grammar engineering system and briefly summarizing the formal underpinnings associated with this choice (§2). We then present a concise summary of the typological literature on clausal complementation (§3), and describe our analysis and implementation in §4. The evaluation

Proceedings of the Society for Computation in Linguistics (SCiL) 2019, pages 39-49.
New York City, New York, January 3-6, 2019

results featuring held-out languages from different families are given in §5. All of the resources mentioned in the paper are freely available.[1]

2 Background

2.1 The Grammar Matrix

We develop a cross-linguistic analysis of clausal complements as part of the LinGO Grammar Matrix[2] (Bender et al., 2002, 2010). Other examples of multilingual grammar engineering projects include ParGram (Butt and King, 2002) and CoreGram (Müller, 2015). What is unique about the Grammar Matrix, however, is that it allows the user to obtain a starter grammar customized from typological choices, making it possible to easily test the analysis on multiple languages, including systematic testing of each part of the analysis using artificially constructed languages (see §4.3).

The Grammar Matrix consists of a web-based questionnaire and a back-end customization system. The input to the system is user answers to the typological questions (elicited in the questionnaire and serialized as a *choices file*) and the output is an implemented starter precision grammar. There are currently multiple libraries, including word order (Bender and Flickinger, 2005; Fokkens, 2014); person, number, gender, and case systems (Drellishak, 2009); tense, aspect, and mood (Poulson, 2011); argument optionality (Saleem and Bender, 2010); matrix yes/no questions, lexicon, (Bender and Flickinger, 2005); morphotactics (O'Hara, 2008; Goodman and Bender, 2010); nominalized clauses (Howell et al., 2018), and clausal modifiers (Howell and Zamaraeva, 2018).[3] The contribution of this paper is the clausal complements library. In the context of the Grammar Matrix, our analysis consists of a new web questionnaire page with a set of choices describing clausal complements cross-linguistically, a set of new types available for the back-end customization logic, and revisions to the types already existing in the system.

Building on the existing resource of the Grammar Matrix to develop our library has dual benefits: On the one hand, we can reuse existing implementations of phenomena that occur in embedded clauses and focus our attention primarily on clause embedding itself. On the other hand, we are able to explore the interaction of our analyses with the existing analyses of other phenomena in order to validate both the existing and new libraries. Finally, because we contribute our library to the Grammar Matrix code base, it can in turn serve as part of the background infrastructure for future library development.

2.2 DELPH-IN Joint Reference Formalism, HPSG and MRS

The Grammar Matrix uses the DELPH-IN Joint Reference Formalism (Copestake, 2000), a version of the HPSG syntactic formalism (Pollard and Sag, 1994), developed to balance expressive power with computational efficiency. Every part of the grammar (lexical items, lexical rules, and phrase structure rules) is encoded with typed feature structures.[4] Phrase structure rules can have any fixed number of daughters in a fixed order,[5] but are typically either binary branching or unary constructions. Lexical rules are always unary projections and are furthermore restricted to the lower parts of the tree, i.e. below any phrase structure rules.

A customized grammar consists of a type hierarchy where typed feature structures (*types*) inherit constraints from other types and add their own constraints. The typed feature structures defined by the grammar are combined using the operation of unification in order to build analyses of sentences. If no analysis can be found for a string using a grammar, then the string is deemed ungrammatical by that grammar. Because the feature structures encode both syntactic and semantic constraints, any derivation tree produced by the grammar also includes an MRS semantic representation (Copestake et al., 2005). When evaluating a grammar created via the Grammar Matrix customization system, the correctness of both syntactic and semantic representations of strings is taken into account.

2.3 Clausal Complements in HPSG

To our knowledge, there is no previous cross-linguistic account of clausal complements in HPSG. Language-specific analyses include Ginzburg and Sag 2000 for English and Crysmann 2013 for German, among others. We could not di-

[1] `svn://lemur.ling.washington.edu/shared/matrix/trunk`, revision 42067 (code and testsuites).

[2] `http://matrix.delph-in.net/customize/matrix.cgi`

[3] The nonexhaustive list of cited libraries includes the ones with which we tested the interaction of our library.

[4] These are exemplified in §4.2 in (4)-(10).

[5] This contrasts with other versions of HPSG formalisms which separate immediate dominance from linear precedence (e.g. Engelkamp et al., 1992) as well as those that allow variable-arity rules, like Sag et al. 2003.

rectly incorporate them for two reasons. First, most of the existing theoretical analyses are not within the DELPH-IN JRF. Second, the literature mostly concerns itself with the level of clause complexity which is beyond the Grammar Matrix's current scope. The analyses which informed us the most are actual grammar implementations within the DELPH-IN JRF and include Siegel et al. 2016 for Japanese and Flickinger 2000, 2011 for English.

3 The Typology of Clausal Complements

Clausal arguments are clauses which serve as core arguments to verbs (Noonan, 2007). They can be subjects or complements (objects) and typically occur with verbs of thought, perception, knowledge, etc. An example of an objectival clausal argument is given in (1).

(1) Kim thinks [that Sandy left]. [eng]

Clausal complements can be finite clauses that can occur on their own (1) or they can be *reduced*. Reduced clauses can share the subject with the matrix clause, have the verb in a dependent form or have case marking on the arguments that differs from what is normal for finite clauses in the language (*ibid.*). Complementation strategies include parataxis (several clauses joined without subordination); complementizer + finite complement; nominalization; infinitival and participial complements.

Here we focus on the complementation phenomena which involve embedded clauses that are declarative, objectival and clearly marked as subordinate, and that contain a full proposition. Embedded clauses that are subjects, interrogative clauses, subject sharing, paratactic, infinitival or participial complements are not covered at this point. Our library supports clausal complements marked by special morphology on the verb, special word order, and/or a complementizer attaching to one of the edges of the embedded clause. Special morphology on the embedded verb is illustrated in (2), a Turkish example with nominalization. Special word order in the matrix clause[6] is illustrated by (3): Uzbek is a SOV language, but clausal complements can be extraposed to the end of the sentence. Note that the order of the complementizer and the embedded clause here also does not follow the same rule as, for example, a transitive verb and its nominal object. Complementizers can be seen in both (3) and (1).

(2) [senin sinema-ya gel-me-n-i]
[2SG.GEN cinema-DAT come-NMZ-2SG-ACC]
isti-yor-um
want-PROG-1SG
'I want you to come to the movies.' [tur] (adapted from Kornfilt 2013, p. 48)

(3) Men bilamen [ki bu ɔdam ǰoǰa-ni
I know-1SG [COMP this man chicken-OBJ
oğirladi]
stole-3SG]
'I know that the man stole the chicken.' [uzb] (Noonan, 2007)

4 Developing the Library

Development starts with mapping typological descriptions of the phenomenon to a set of choices to be presented to the user via the web questionnaire (§4.1). This characterizes our analysis in the broad sense. In particular, upon reviewing the typological literature and restricting the scope of the library as outlined in §3, we provide an analysis for languages that mark clausal complements in a set of certain ways, e.g. with complementizers, nominalization, aspect or other marking on the embedded verb.[7] We posit HPSG analyses for all combinations of the choices we target, without claiming to have identified or modeled all dimensions of variation. We start with basic analyses for complementizers and clause-embedding verbs (§4.2) and implement a holistic analysis in a test-driven fashion (§4.3).

4.1 Web Questionnaire

As summarized in §3, clausal complements can be marked with a complementizer, special morphology on the embedded verb, or word order changes in either the matrix or the embedded clause. Combinations of these phenomena are also possible. There may be several distinct strategies in a language and clausal complement-taking verbs may select for specific ones. We add a web page to the questionnaire which elicits all the relevant choices from the user.

A language may have multiple complementizers which belong to distinct complementation strategies (i.e. with different word order consequences, different selecting verbs, or different morphological requirements on the embedded verb). To ac-

[6] See §4.2.4 regarding the word order in subordinate clauses in German.

[7] While our analysis draws on a comprehensive review of typological literature, we do not make any typological claims as part of this work.

count for this, we generalize the FORM feature, previously available for verbs to distinguish different syntactically relevant inflected forms, to complementizers, and reflect this in the questionnaire.

To handle complementation strategies that require specific morphology on the embedded verb, we leverage several existing libraries: The morphotactics library (Goodman, 2013) provides a means of associating morphological features with inflected forms of verbs; the tense/aspect/mood library (Poulson, 2011) allows users to define morphology associated with those features (as well as additional arbitrary features on the 'Other Features' page) and the nominalization library (Howell et al., 2018) accounts for nominalized clauses. In all cases, the lexical rules defined by these libraries include morphosyntactic or morphosemantic features which allow an embedding verb or complementizer to select for a clausal complement headed by a verb with appropriate morphology. This lexically introduced information is available at the clause level thanks to constraints in the shared core grammar that pass the information up the tree.

We also provide an analysis of clause-bounded extraposition of clausal complements. One of the salient syntactic characteristics of clausal complements is that they tend to be dispreferred in sentence-medial position, as seen in (3). We allow the user to choose this type of extraposition and indicate whether it is obligatory or optional.[8]

Finally, the user must add lexical types for clause-embedding verbs, at least one per complementation strategy, as we operate within a lexicalist view of syntax and the lexical type of the clause-embedding verb is therefore responsible for choosing which type of clausal complement to take. Here, we extended an already existing part of the questionnaire (the Lexicon) so that verb types can select for a complementation strategy.

4.2 Syntactic Analysis

As discussed in §2.3, there appears to be little in the theoretical HPSG literature focused on a cross-linguistic account of basic complementation that can be directly incorporated into our library. However, we can leverage the existing Grammar Matrix analyses of various phenomena as well as those from other implemented grammars and extend or adapt them to cover clausal complementation; the Grammar Matrix provides us with a theoretical as well as an engineering basis. As is consistent with the Grammar Matrix's overarching goal, the main goal of our analysis is to provide the user-linguist with a range of typologically motivated possibilities rather than to focus on specific predictions of what is possible vs. what is not possible in the world's languages. In this section, we describe the building blocks of our analysis.

4.2.1 Lexical Types

Complementizers can be treated as semantically empty elements that take a proposition as a complement (e.g. Siegel et al. 2016). The Grammar Matrix already had a complementizer type which did not contribute its own predication but raised its complement's semantic identifier via the HOOK identity, as shown in (4).[9]

$$\text{(4)}\quad \begin{bmatrix} \textit{comp-lex-item} & \\ \text{HEAD} & \begin{bmatrix} \textit{comp} & \\ \text{MOD} & \langle\,\rangle \end{bmatrix} \\ \text{SUBJ} & \langle\,\rangle \\ \text{COMPS} & \left\langle \boxed{1} \begin{bmatrix} \text{HEAD} & \textit{verb} \\ \text{SUBJ} & \langle\,\rangle \\ \text{COMPS} & \langle\,\rangle \\ \text{HOOK} & \boxed{2} \end{bmatrix} \right\rangle \\ \text{ARG-ST} & \langle \boxed{1} \rangle \\ \text{RELS} & \langle !\ ! \rangle \\ \text{HOOK} & \boxed{2} \end{bmatrix}$$

While posited originally for question particles in the yes/no questions library (Bender and Flickinger, 2005), it was intended to generalize to clausal complementizers. In order to have clausal complementizers inherit from this supertype we move the MC ('main clause') value from the supertype (4) where it was posited originally to the clausal complementizer subtypes (5). This way question particles (not shown) can take main clauses as their complements but clausal complementizers cannot. Another change implemented to support our analysis is that complementizers now have the FORM feature (with user-specified array of values). This is used to license certain complementizers only in combination with certain verb types. [10]

[8] Displacement to sentence-initial position is less mentioned in the typological literature, though it appears to be possible. We leave it to future work.

[9] Empty semantic relations (RELS) lists and HOOK identities as in (4) are characteristic of semantically empty elements.

[10] AVMs are abbreviated to focus on the primary points of interest. Note also that all such feature structures are arranged into a type hierarchy, such that more specific types, like (5), inherit all of the constraints of their supertypes, like (4).

(5) $$\begin{bmatrix} \textit{ccomp-lex-item} & \\ \text{HEAD} & \begin{bmatrix}\text{FORM} & \textit{form}\end{bmatrix} \\ \text{COMPS} & \left\langle \begin{bmatrix}\text{MC} & -\end{bmatrix} \right\rangle \end{bmatrix}$$

To model clausal complement-taking verbs,[11] we introduce new lexical types. Lexical types in the Grammar Matrix typically specify both syntactic and semantic requirements on their dependents. Because cross-linguistically clausal complement-taking verbs can take both verbal and nominalized complements, our supertype in this space (6) leaves both underspecified. Otherwise, it is similar to *transitive-verb-lex*, which also specifies two arguments.[12]

(6) $$\begin{bmatrix} \textit{cl-verb-lex} & \\ \text{SUBJ} & \langle \boxed{1} \rangle \\ \text{COMPS} & \left\langle \boxed{2} \begin{bmatrix}\text{SUBJ} & \langle\,\rangle \\ \text{COMPS} & \langle\,\rangle\end{bmatrix} \right\rangle \\ \text{ARG-ST} & \left\langle \boxed{1} \begin{bmatrix}\text{HEAD} & \textit{noun}\end{bmatrix}, \boxed{2} \right\rangle \end{bmatrix}$$

Further subtypes inheriting from *cl-verb-lex* are based on the specific choices made by the user and make use of one of two already existing types: either *transitive-lex-item*, for nominalized complements, or *clausal-second-arg-trans-lex-item*, for all other types of clausal complements. This allows us to model the primary semantic effect of nominalization: the introduction of an *individual*-type variable corresponding to the nominalized clause, which in turn allows for e.g. adjectival modifiers of that clause. In the semantic composition, *transitive-lex-item* takes the *individual*-type index of its complement as a semantic argument. *Clausal-second-arg-trans-lex-item*, on the other hand, expects an argument with an *event*-type index and combines with it semantically via a handle constraint to accommodate the MRS analysis of quantifier scope (Copestake et al., 2005).

For proper interaction with the case library (Drellishak, 2009), clausal complement-taking verbs require a hierarchy which is aware of various case frames. In modeling a language with case, the user must specify a case frame for each verb type. For example, a verb can have nominative-accusative argument structure, meaning the type will constrain its syntactic subject to bear nominative case and its object accusative. We added clausal complement strategies as argument structure choices and allowed the case frames that are available for non-clause-embedding verbs to be available for clausal complement strategies as well. The option which specifies a case constraint on the verb's object is not always available: it only makes sense for nominalized clausal objects.

4.2.2 Lexical Rules, Features, Nominalization

As noted in §4.1 above, we leverage existing libraries to account for constraints on the morphology of embedded verbs imposed by complementizers or clausal-complement verbs. The result is that users can specify, for example, that the clausal complement be nominalized, at which point the resulting clausal-complement verb type will include the constraint shown in (7). We also extended the Morphotactics library to allow nominalized verbs to bear case inflections.

(7) $$\begin{bmatrix} \textit{cl-verb-lex} & \\ \text{COMPS} & \left\langle \begin{bmatrix}\text{NMZ} + \end{bmatrix} \right\rangle \end{bmatrix}$$

For a specific example of how we accommodate lexical variation in complementation, consider Turkish, which has several complementation strategies. The verb *isti* ('want') can not only take nominalized complements like in (2) but also clauses headed by verbs in the optative form, as in (8).

(8) herkes [yarin ben-im-le sinema-ya
everybody [tomorrow I-GEN-WITH cinema-DAT
gel-esin] isti-yor
come-2SG.OPT] want-PRES.PROG
'Everybody wants you to come along to the movies with me tomorrow.' [tur] (from Kornfilt 2013, p. 48)

To model this, the user can add separate complementation strategies in the questionnaire for nominalization and for optative on the embedded verb, and a corresponding clausal complement-taking verb type to go with each strategy. The resulting grammar will include two lexical entries for *isti*, each belonging to a different type. One will only take complements with [NMZ +, FORM nonfinite] and the other [NMZ −, FORM opt].

4.2.3 Phrase Structure Rules, INIT and EXTRA

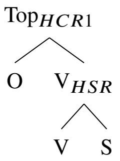
The most intricate part of this library is its interaction with the existing analysis of word order. To account for basic word order, any Grammar Matrix-generated grammar will have some basic head-subject and head-complement rules, one of each when the word order is strict (e.g. SOV) and more if the order is flexible.

[11] Often called the CTP, complement-taking predicate, in syntactic and typological literature.

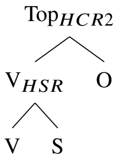
[12] In future work, we will extend the lexical types available to provide for more valence patterns with clausal complements, such as *Kim told me that Sandy left.*

When the complementizer or the main verb participates in a word order that is different from other verbs, we posit an additional head-complement rule (HCR) and, in some cases, an additional head-subject rule (HSR). Then we constrain them (as well as all lexical types that use the HCRs and HSRs) for one or both Boolean features, INIT and EXTRA. While similar features have been used before (Keller 1995; Crysmann 2013; see also Siegel et al. 2016, 59), we incorporate them into a new customization logic so that sets of correct constraints are emitted automatically based on user choices. Examples (9)–(10) illustrate the general HCR and the additional one for extraposition that are emitted for one of many possible user-defined languages: an OVS language with extraposition.

(9)
$$\begin{bmatrix} \textit{HCR1} & \\ \text{COMPS} & \boxed{1} \\ \text{H-DTR} & \begin{bmatrix} \text{INIT} & - \\ \text{SUBJ} & \langle\,\rangle \\ \text{COMPS} & \langle\boxed{3}\rangle \oplus \boxed{1} \end{bmatrix} \\ \text{NH-DTR} & \boxed{3} \\ \text{ARGS} & \langle\,\boxed{3}\,,\boxed{2}\,\rangle \end{bmatrix}$$



(10)
$$\begin{bmatrix} \textit{HCR2} & \\ \text{COMPS} & \boxed{1} \\ \text{H-DTR} & \begin{bmatrix} \text{INIT} & + \\ \text{SUBJ} & \langle\,\rangle \\ \text{COMPS} & \langle\boxed{3}\rangle \oplus \boxed{1} \end{bmatrix} \\ \text{NH-DTR} & \boxed{3} \\ \text{ARGS} & \langle\,\boxed{2}\,,\boxed{3}\,\rangle \end{bmatrix}$$



We use INIT (a HEAD feature) on lexical types and on the head daughters of phrase structure rules to account for word order variations associated with subordinator attachment and with extraposition of objects from OV languages. [INIT +] is associated with head-initial rules and [INIT −] with head-final rules. We need a separate feature, EXTRA (not shown), to handle extraposition in VOS and V-initial languages, because the order in these languages remains head-initial despite extraposition to the end of sentence. EXTRA is constrained on COMPS list elements of clausal-complement selecting heads and on the non-head daughter of head-initial HCRs.[13]

4.2.4 German-like V2/V-final Variation

In German, the word order is verb second (V2), but in clausal complements marked by a complementizer the order is verb final. Fokkens (2014) implemented this type of word order variation in the Grammar Matrix framework but did not fully integrate it into the customization system. We incorporate part of her analysis so that the user can choose this type of word order variation directly via the questionnaire.

4.3 Implementation

Encoding any particular structure is straightforward; the challenge is in emitting the right sets of structures with the right constraints given user choices, where the space of possible combinations is large. Furthermore, any additions need to be well integrated so that the constraints the new code emits should not interfere with what other libraries create. At the same time, it is not desirable to add unique structures where an existing one can be adapted, since this leads to unnecessarily large, complex and potentially overgenerating grammars. The goal is to implement an analysis that is sufficiently general to handle any typologically plausible language while at the same time emitting reasonably streamlined grammars.

4.3.1 Test-driven Development

We begin by describing the procedure we use for creating test cases, both in development and in evaluation (§5). Test-driven development in the context of the Grammar Matrix relies on two components: the testsuites and the test choices files (i.e. grammar specifications). There is a testsuite and a choices file for each language that is used in development or evaluation. While the testsuite and the choices file are created separately, both are based on the descriptive grammar for the language in question (with the exception of artificial pseudolanguages discussed below which function more like ‘unit tests’ for the bits of our analysis). Specifically, we first collect the sentences illustrating clausal complementation from the descriptive grammar to compile the testsuites. Then, in a separate iteration, we read the descriptive grammar to the extent necessary to fill out the Grammar Matrix questionnaire for this language.

We build testsuites and choices files for *pseudolanguages* (defined by combinations of choices) and for 5 development languages (see §5 for the details). The number of possible pseudolanguages is large: a conservative estimate which treats some bundles of choices as single dimensions is 2300, assuming one strategy per language. In testing we

[13] For an example of how EXTRA is used, see Zamaraeva et al. 2018.

work with a 50 language sample which includes several languages with more than one complementation strategy. A testsuite consists of grammatical and ungrammatical sentences illustrating what is possible/impossible with respect to clausal complements in a given language.

For pseudolanguages testsuites, we construct all possible nonrecursive sentences[14] illustrating the clausal complementation strategies that this language has. For development languages, the sentences either come directly or are adapted from the sections in descriptive grammars explaining how clausal complements work in this language. Here the sentences feature interacting phenomena such as tense and case.[15] Crucially, we include corresponding impossible (ungrammatical) sentences to make sure they are not parsed. Consider a language with one strict strategy: SOV word order, obligatory complementizer attaching before the embedded clause, and obligatory extraposition of clausal complements. The testsuite will consist of 2 grammatical sentences, one for a simple SOV sentence and another that has an extraposed clausal complement with the complementizer. The ungrammatical sentences will include: a complex sentence with no complementizer, one with a complementizer attaching after the clause, one with a non-extraposed clausal complement, and a simple clause with SVO order. The more flexible the language, the more grammatical and fewer ungrammatical examples the testsuite will have.

From the choices files which are created independently from the testsuites, grammar fragments are created with the original customization system. The grammars are loaded into the LKB software(Copestake, 2002) which can parse strings. Then we edit these grammars by hand until they behave correctly with respect to the yet unsupported clausal complements choices (as reflected by the grammars' coverage and overgeneration over the corresponding testsuites). Then we generalize the solutions in the resulting grammars and incorporate them in the customization system, continually testing with the regression tests. The result of this development is frozen before evaluation on held-out languages, described in §5 below.

4.3.2 Adding Types

We update the customization system as follows. A supertype for clausal-complement verbs (described in §4.2) is now added in all cases when any clausal complement strategy is specified. A supertype for a complementizer is added whenever the user says there is a complementizer associated with any of the strategies. Appropriate subtypes are added based on the user choices, one per strategy, and constrained as described in §4.3.3.[16]

Additional phrase structure rules are added according to the user choices. An additional HSR is added for VOS orders with extraposition. An additional HCR is added whenever the complementizer or the clause-embedding verb cannot use the basic HCR. In general, this means all situations when the basic order is e.g. head-final but there is either extraposition or the complementizer can attach clause-initially (or symmetrically, if the basic order is head-initial but the complementizer can attach clause-finally). In practice, it means checking combinations of choices: 8 for the word orders;[17] 3 for complementizer (obligatory, no, optional); 3 for complementizer attachment (before the clause, after, or both); 3 for extraposition (obligatory, no, optional).

4.3.3 Adding Constraints

After adding the types, we constrain them for the grammar to generate only grammatical strings (e.g. for features FORM, NMZ, INIT, EXTRA). We need to add constraints so that they do not clash with those placed by other libraries while not positing new types unless necessary.

The information about whether FORM or NMZ (nominalization) constraints are needed on the relevant types comes directly from the choices files: the user would have specified a FORM value or a nominalization strategy associated with the complementation strategy. The treatment of INIT and EXTRA however must be inferred from a fairly large space of choices combinations.

Figure 1 shows the logic that we add to the customization system that lets it decide whether to use the INIT feature.[18] The decision depends on

[14] The vocabulary is minimal.

[15] In developing these testsuites, we sometimes must adapt the sentences to exclude phenomena that are not supposed to be supported by the system or to capture the full spectrum described by the grammar's author in prose but not fully illustrated by examples. This, along with the initial selection of which sentences to include in the grammar, introduces a certain amount of bias to the testsuites.

[16] A type can be instantiated by multiple lexical entries.

[17] Excluding V2 and free.

[18] Abbreviations for Figure and Tables: comp (complementizer); extrap (extraposition); fam (family); morph (morpheme); neg (ungrammatical sentences); nmz (nominalization); oblig (obligatory); opt (optional); OV (object pre-

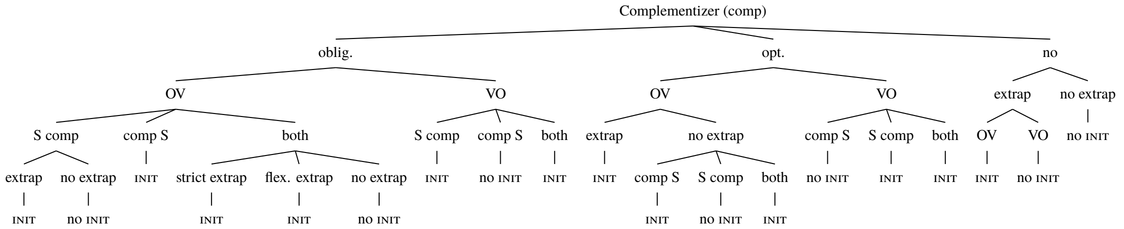


Figure 1: Decision tree illustrating the logic of using the INIT feature based on user choices

whether there is a complementizer, whether it is obligatory or optional, what is the basic word order, is there extraposition and is it strict or flexible, and how does the complementizer attach to the clause. If the INIT feature is used, then all HCR and all lexical items which can go through the HCR must be properly constrained for INIT + or −. The logic associated with the EXTRA feature is simpler: it is used only with VOS and V-initial word orders when there is extraposition.

4.3.4 Summary

This section described the process and the main techniques used to implement the Clausal Complements library for the Grammar Matrix. We operate in a large space of theoretically possible user choices combinations (over 2300) where the ultimate goal is to emit streamlined grammars which behave correctly for any valid combination of choices (and a language defined by them). The development was driven by a sample of pseudo- and illustrative (development) languages and was evaluated as described in the next section.

5 Testing, Evaluation, and Error Analysis

To test the typological legitimacy and the rigor of our analysis, we apply a three-stage process where we test grammars against testsuites. The process used in all stages is described in §4.3.1. The first two stages involving 50 pseudolanguages and 5 development languages are part of the test-driven development. We chose the development languages which exhibited the full range of in-scope complementation phenomena and for which we had both some sentences and a description sufficient for us to fill out the questionnaire. Table 1 summarizes the phenomena covered by the development languages.[19]

As for testing the interaction with other libraries, we included all the choices that were relevant to the testsuites. For example, if the language has case and the sentences with clausal complements showed this, we filled out the Case portion of the questionnaire and added appropriate lexical rules on the Morphology page. Case, word order, morphology, and tense, aspect and mood choices came up most often.

After addressing a few issues uncovered by the pseudo- and development languages, we achieve 100% coverage and 0% overgeneration on their testsuites (Table 1). Furthermore, we checked that all parses for each sentence are warranted, i.e. there is no unwanted ambiguity. At the same time, the library was used in a course where students implemented grammar fragments for 5 more languages,[21] which helped us discover an issue in the interaction with the information structure library (Song, 2014).

Finally, we evaluate the library on 5 held-out languages from different language families than the development languages.[22] The results are presented in Table 2. Coverage and overgeneration are given in numbers of sentences in the testsuite (4/8 means 4 out of 8 were parsed).

In Jalkunan (SOV), there is a kind of clausal

cedes verb); pos (grammatical sentences); S comp (clause followed by complementizer); strat (strategy); VO (verb precedes object); WO (word order). Language families: AA (Afro-Asiatic); Astrn (Austronesian); Awk (Arawak); IE (Indo-European); NS (Nilo-Saharan); NC (Niger-Congo); PN (Pama-Nyungan); PP (Plateau Penutian); Tur (Turkic).

[19] German (Sapp, 2006), Tagalog (Hoenigswald et al., 1975), Lango (Noonan, 2007), Turkish (Kornfilt, 2013). The first author, a native speaker of Russian, provided the testsuite and the choices for Russian.

[20] We targeted a fragment of German with obligatory complementizers and V2/V-final word order variation.

[21] Akuntsu [aqz] (Tupian); Lhasa Tibetan [bod] (Tibeto-Burman), Dzongkha [dzo] (Tibeto-Burman), Dagaare [gur] (Niger-Congo), Yolmo [scp] (Tibeto-Burman).

[22] The number of languages is limited by time constraints. We pick held-out languages randomly from a pool of descriptive grammars and reject them if they come from a language family already used or if the grammars do not cover basic clausal complements. Grammars: Jalkunan (Heath, 2017), Paresi-Haliti (Brandão), Sahaptin (Jansen, 2010), Hebrew (Zuckermann, 2006), Wangkangurru (Hercus, 1994).

Language	iso639-3	fam	WO	comp	order	morph	extrap	# strat	pos	neg
Russian	rus	IE	free	opt	comp S	nmz,form	-	3	6	11
German	deu	IE	V2/V-fin	oblig[20]	comp S.	-	-	1	6	4
Tagalog	tgl	Astrn.	V-in	oblig	comp S	-	flexible	1	3	4
Lango	laj	NS	SVO	oblig	comp S	mood	-	3	4	4
Turkish	tur	Tur	SOV	opt	both	nmz, form	strict	4	7	9

Table 1: Phenomena covered in the development languages. Coverage is 100%, overgeneration is 0%.

Language	iso639-3	fam	WO	comp	order	morph	extrap	# strat.	Cov.	Overgen.
Jalkunan	bxl	NC	SOV	opt	comp S	-	strict	1	4/8	0/12
Paresi-Haliti	pab	Awk	SOV	-	-	nmz	strict	1	5/5	0/5
Yakima Sahaptin	yak	PP	free	-	-	nmz	-	1	10/10	0/6
Modern Hebrew	heb	AA	SVO	oblig	comp S	-	-	1	2/2	0/9
Wangkangurru	wgg	PN	free	-	-	aspect	-	1	10/10	0/3

Table 2: Coverage and overgeneration on sentences from held-out language families.

complement extraposition where a "dummy" 3sg pronoun (similar to *it* in English) stays in the sentence-medial position while the clausal complement is extraposed (Heath, 2017). We had not considered this strategy during the development since we hadn't come across it in the literature, so we could not parse those items. In addition, we discovered a bug in the interaction with the word order library which places constraints on the order of auxiliaries and their complements. Some of the Jalkunan sentences had auxiliaries, and we also could not parse them due to the bug; hence the 50% coverage. On all other languages we achieve 100% coverage and 0% overgeneration (again, with no spurious ambiguity).

The testsuites represent typologically diverse languages, and so one testsuite may emphasize word order variation, another morphology, etc. The descriptive grammars for held-out languages only included one in-scope strategy per language. In general, our held-out grammars contain relatively few examples which illustrated basic complementation.[23] Our source for Modern Hebrew, Zuckermann 2006, for example, identified a single strict complementation strategy, and so for that language we have just one grammatical sentence illustrating simple noun complementation and one grammatical sentence illustrating clausal complementation. The more strategies in a language, the more flexible they are, and the clearer the author of the grammar describes them in isolation from other phenomena, the more sentences we can include. The point of the evaluation is to approximate quantitatively how well a system would do for a linguist working from a reasonably thorough description of a language we had not worked with.

6 Conclusion

We presented a clausal complements library for the LinGO Grammar Matrix. To this end, we implemented an HPSG-based analysis of a range of complementation phenomena attested in the typological literature. The library allows the grammar engineer to include basic clausal complements in their starter grammar in a streamlined system with appropriate interaction with other phenomena, thus extending the initial coverage of the grammar fragment that can be obtained automatically. Available choices define thousands of possible combinations for which the library needs to emit correct code. We make sure we have correct behavior on a sample of 50 choices combinations and 5 development languages; on testsuites from 5 held-out languages, we achieve 88% coverage and no overgeneration.

In the future, we plan to extend the system to handle such related phenomena as subject sharing, sentential subjects, embedded questions, and displacement to the beginning of the clause. In terms of interactions with other libraries, we will look at complex sentences combining clausal complements and clausal modifiers and coordinated clausal complements. Our library will also serve as a foundation for testing future additions to the Grammar Matrix for their interaction with subordinate clauses, e.g. *wh*-questions. Finally, we plan to explore how to infer answers to our additions to the questionnaire from collections of interlinear glossed text, along the lines of Bender et al. (2013) and Bender et al. (2014).

[23] Descriptive grammars seem to prioritize complicated examples of complex clauses. This differs from the data from development languages which were picked so as to better inform the implementation of the library.

Acknowledgements

We thank the anonymous reviewers for SCiL 2019 for helpful discussion.

This material is based upon work supported by the National Science Foundation under Grant No. BCS-1561833. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Emily M. Bender, Joshua Crowgey, Michael Wayne Goodman, and Fei Xia. 2014. Learning grammar specifications from IGT: A case study of Chintang. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 43–53. Association for Computational Linguistics.

Emily M Bender, Scott Drellishak, Antske Fokkens, Laurie Poulson, and Safiyyah Saleem. 2010. Grammar customization. *Research on Language & Computation*, 8(1):23–72. 10.1007/s11168-010-9070-1.

Emily M. Bender and Dan Flickinger. 2005. Rapid prototyping of scalable grammars: Towards modularity in extensions to a language-independent core. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing IJCNLP-05 (Posters/Demos)*, Jeju Island, Korea.

Emily M. Bender, Dan Flickinger, and Stephan Oepen. 2002. The Grammar Matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pages 8–14, Taipei, Taiwan.

Emily M. Bender, Michael Wayne Goodman, Joshua Crowgey, and Fei Xia. 2013. Towards creating precision grammars from interlinear glossed text: Inferring large-scale typological properties. In *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 74–83, Sofia, Bulgaria. Association for Computational Linguistics.

Ana Paula Barros Brandão. *A Reference Grammar of Paresi-Haliti (Arawak)*. Ph.D. thesis, The University of Texas at Austin.

Miriam Butt and Tracy Holloway King. 2002. Urdu and the Parallel Grammar project. In *Proceedings of the 3rd workshop on Asian language resources and international standardization-Volume 12*, pages 1–3. Association for Computational Linguistics.

Ann Copestake. 2000. Appendix: Definitions of typed feature structures. *Natural Language Engineering*, 6(01):109–112.

Ann Copestake. 2002. *Implementing typed feature structure grammars*. CSLI publications Stanford.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A Sag. 2005. Minimal recursion semantics: An introduction. *Research on language and computation*, 3(2-3):281–332.

Berthold Crysmann. 2013. On the locality of complement clause and relative clause extraposition. *Rightward movement in a comparative perspective*, 200:369–395.

Scott Drellishak. 2009. *Widespread but Not Universal: Improving the Typological Coverage of the Grammar Matrix*. Ph.D. thesis, University of Washington.

Judith Engelkamp, Gregor Erbach, and Hans Uszkoreit. 1992. Handling linear precedence constraints by unification. In *Proceedings of the 30th annual meeting on Association for Computational Linguistics*, pages 201–208. Association for Computational Linguistics.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(01):15–28.

Dan Flickinger. 2011. Accuracy v. robustness in grammar engineering. In Emily M. Bender and Jennifer E. Arnold, editors, *Language from a Cognitive Perspective: Grammar, Usage and Processing*, pages 31–50. CSLI Publications, Stanford, CA.

Antske Sibelle Fokkens. 2014. *Enhancing Empirical Research for Linguistically Motivated Precision Grammars*. Ph.D. thesis, Department of Computational Linguistics, Universität des Saarlandes.

Jonathan Ginzburg and Ivan Sag. 2000. *Interrogative investigations*. Stanford: CSLI publications.

Michael Wayne Goodman. 2013. Generation of machine-readable morphological rules from human-readable input. *Seattle: University of Washington Working Papers in Linguistics*, 30.

Michael Wayne Goodman and Emily M. Bender. 2010. What's in a word? Refining the morphotactic infrastructure in the LinGO Grammar Matrix customization system. In *Workshop on Morphology and Formal Grammar, Paris*.

Jeffrey Heath. 2017. *A grammar of Jalkunan (Mande, Burkina Faso)*. Language Description Heritage Library.

Luise A Hercus. 1994. *A grammar of the Arabana-Wangkangurru language: Lake Eyre Basin, South Australia*. Australian National Univ.

Henry M Hoenigswald, Paul Schachter, and Fe T Otanes. 1975. *Tagalog Reference Grammar*.

Kristen Howell and Olga Zamaraeva. 2018. Clausal modifiers in the grammar matrix. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2939–2952.

Kristen Howell, Olga Zamaraeva, and Emily M. Bender. 2018. Nominalized clauses in the Grammar Matrix. In *Proceedings of the 25th International Conference on Head-Driven Phrase Structure Grammar*, pages 68–88.

Joana Worth Jansen. 2010. *A Grammar of Yakima Ichishkíin/Sahaptin*. Ph.D. thesis, University of Oregon.

Frank Keller. 1995. Towards an account of extraposition in HPSG. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, pages 301–306. Morgan Kaufmann Publishers Inc.

Jaklin Kornfilt. 2013. *Turkish*. Routledge, London.

Stefan Müller. 2015. The CoreGram project: Theoretical linguistics, theory development and verification. *Journal of Language Modelling*, 3(1):21–86.

Michael Noonan. 2007. Complementation. In Timothy Shopen, editor, *Language Typology and Syntactic Description*, volume 2. Cambridge University Press, Cambridge, UK.

Kelly O'Hara. 2008. *A morphotactic infrastructure for a grammar customization system*. Ph.D. thesis, University of Washington.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press and CSLI Publications, Chicago, IL and Stanford, CA.

Laurie Poulson. 2011. Meta-modeling of tense and aspect in a cross-linguistic grammar engineering platform. *University of Washington Working Papers in Linguistics (UWWPL)*, 28.

Ivan A. Sag, Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*, second edition. CSLI, Stanford, CA.

Safiyyah Saleem and Emily M Bender. 2010. Argument optionality in the LinGO Grammar Matrix. In *Proceedings of the 23rd international conference on computational linguistics: posters*, pages 1068–1076. Association for Computational Linguistics.

Christopher D Sapp. 2006. *Verb order in subordinate clauses from Early New High German to Modern German*. Indiana University.

Melanie Siegel, Emily M. Bender, and Francis Bond. 2016. *Jacy: An Implemented Grammar of Japanese*. CSLI Studies in Computational Linguistics. CSLI Publications, Stanford CA.

Sanghoun Song. 2014. *A grammar library for information structure*. Ph.D. thesis, University of Washington.

Olga Zamaraeva, Kristen Howell, and Emily M Bender. 2018. A cross-linguistic account of subordinator and subordinate clause position. Poster presented at The 25th International Conference on Head-Driven Phrase Structure Grammar, Tokyo, Japan.

Ghil'ad Zuckermann. 2006. Complement clause types in Israeli. Oxford University Press.