The Syntactic Exponence of Sentential Negation: a model for the LinGO Grammar Matrix

Joshua Crowgey

A thesis
submitted in partial fulfillment of the
requirements for degree of

Master of Arts

University of Washington

2012

Committee:

Emily M. Bender

Edith Aldrige

Program Authorized to Offer Degree:
Linguistics

TABLE OF CONTENTS

ACKNOWLEDGMENTS

This thesis would have never been started (much less completed) without the support of my family and my loving partner. Lisa, Jason, Mom and Dad: thank you for believing in me when I didn't believe in myself.

I also owe a deep debt of gratitude to my committee, Emily M. Bender and Edith Aldrige. Emily, my advisor, introduced me to the fascinating study at the nexus of linguistic typology and implemented syntax and I am continually grateful to her for inviting me into this pursuit. I want to thank Edith Aldridge for encouraging me to explore the comparison between different linguistic theories. I owe both these linguists special thanks for patiently reading vaguely worded and poorly formatted drafts and then pointing the way forward.

To the Grammar Matrix development group and to the larger DELPH-IN community I also owe a debt of gratitude, not only for helping me learn to use the shared tools to debug my grammars, but for your patience explaining things to me again and again until I understood.

I want to express sincere thanks to Joyce Parvi and Mike Furr of the Linguistics Department. Joyce was my first contact with academic linguistics at UW and she helped me successfully navigate the daunting and frighteningly bureaucratic process of joining a large research institution as a graduate student. Joyce, I really don't think I could have done it without you. Mike provides good humor and engaging prompts about my work every time I see him.

Furthermore, I owe thanks to each of the instructors I have sat under in the Linguistics Department. It's not only because of the opportunities you have provided me, but also because of your inspirational research and teaching that I continue to pursue my studies in this field.

To my good friend Sanjay Rao, thanks for always casting my linguistic problems in

a unique light. For countless fascinating discussions and patient explanations of topics linguistic and other: I owe you a large debt of gratitude.

Finally, I have to thank Bird (my canine best friend) for emotional support *par excellence*. Her unconditional love buoys my soul.

This material is based upon work supported by the National Science Foundation under Grant No. 0644097. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

DEDICATION

to my parents, John and Sharon Crowgey

Chapter 1

INTRODUCTION

The Grammar Matrix customization system (Bender et al., 2002, 2010a) expedites the process of initial creation of basic HPSG (Pollard and Sag, 1994) grammars for natural languages. The system provides users with a questionnaire about how the language under study handles certain morphosyntactic and syntactico-semantic phenomena and uses this information to assemble a customized machine-readable grammar which is then available for download by the user. The grammar is put together by accessing stored definitions for grammatical objects and including them in a set of output grammar files or by creating customized subtypes of the stored analyses and including them. The output grammar contains a copy of a set of core definitions, the "core grammar", which has definitions for types used in the analyses of putatively universal phenomena (the head-feature principle, for example). Output grammars also contain type definitions generated by a set of "libraries", which provide customizable options for phenomena which that have been identified in typological literature as widespread but not universal (Drellishak, 2009). Refinement of the Grammar Matrix is an ongoing research project. This thesis proposes an extension of the customization system's (Bender and Flickinger, 2005; Bender et al., 2010a) sentential negation library. I review the HPSG literature surrounding sentential negation and lexicalist syntax as well as the relevant typological literature to combine insights from both fields in the creation of a new library for sentential negation.

I begin in Chapter 2 by reviewing some aspects of HPSG which will frame much of the exploration at hand. Then I present some further details about the architecture of the Grammar Matrix customization system. In the first part of Chapter 3, I review the relevant aspects of lexical material in HPSG and in the second part I look at what evidence from typological surveys can bring to bear on a universal model of sentential negation. In Chapter 4, I present a synthesis of the insights from HPSG theory and typology and propose

a more finely articulated typology of sentential negation. After showing the derivation of the typology, I step through the proposed negation types to show analyses for each type. In Chapter 5, I discuss the implementation of this typology as a library for the customization system. This chapter discusses the information that I solicit from users in customizing a negation strategy and the "behind the scenes" process of assembling the grammatical objects need for each type of the model. Chapter 6 presents two evaluations of the proposed library, a set of regression tests and an experiment in using the new library to model sentential negation in held-out languages. Finally, Chapter 7 provides some concluding remarks.

Chapter 2

BACKGROUND

In this chapter, I first present an overview of the Grammar Matrix customization system and a summary of the tasks undertaken in the construction of this thesis. After that, I describe a few important properties of HPSG which are relevant to this exploration. I will first say a few words about the general organization of grammatical objects in this framework and their description as typed feature structures. Then I turn to the Lexical Integrity Hypothesis (Bresnan and Mchombo, 1995), which motivates a primary division for syntactic exponents which is reflected in Matrix-derived grammars and in the model of sentential negation presented in Chapter 4. Afterwards, I introduce the system of semantic representation used in Matrix derived grammars (Minimal Recursion Semantics (Copestake et al., 2005)) and present a universal representation for sentential negation which is used in this thesis.

2.1 The Grammar Matrix

The Grammar Matrix (Bender et al., 2002, 2010a) is a resource for computational linguistics which (along with accompanying DELPH-IN tools[1]) provides an implementation of HPSG that allows cross-linguistic comparison in a number of natural language processing tasks. Part of the Grammar Matrix is a grammar customization system (Bender and Flickinger, 2005; Bender et al., 2010a)—a web application which provides a user-linguist with the ability to describe a language and then download a machine-readable grammar that can be used to parse and generate sentences, do treebanking, explore semantics, etc.

Users interact with the customization system through a questionnaire where they can specify details of the phenomena exhibited by their language. This questionnaire is divided into subpages which pertain to certain subdomains of linguistic analysis. There

[1] http://delph-in.net/

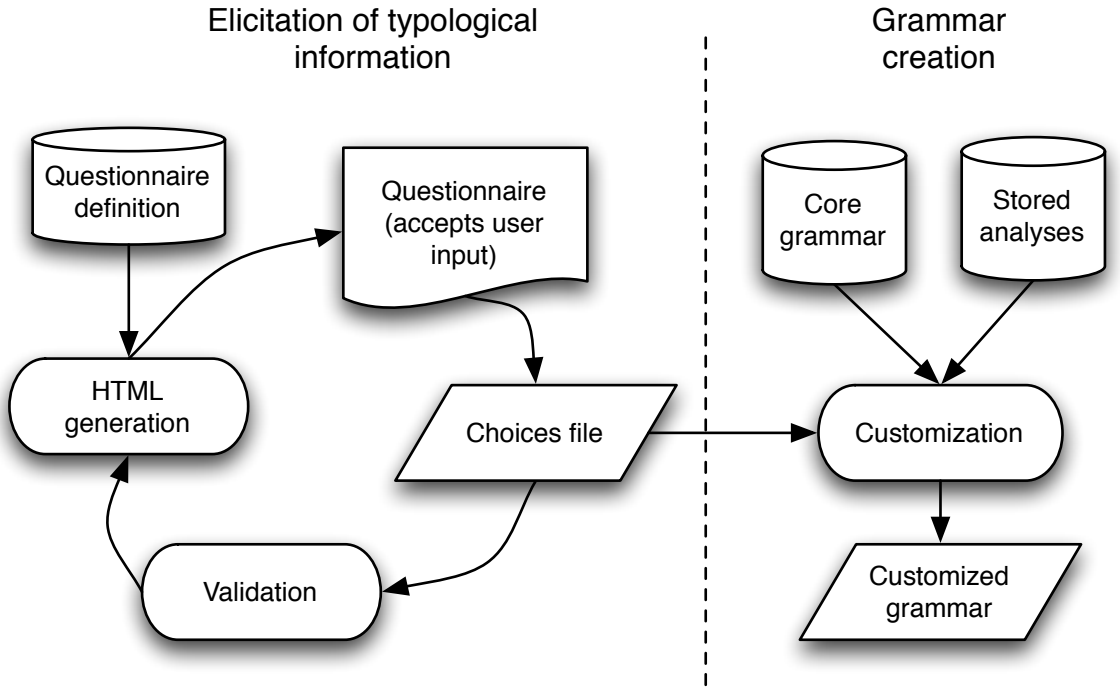
are subpages dedicated to word-order, case marking, argument optionality and sentential negation (amongst others). For each section, the user-linguist is asked to make choices; the options provided are generally drawn from linguistic literature and the development of these options is an ongoing research project. The repository of grammatical types provided by the customization system is divided into those which are putatively universal and those which are common, but not necessarily applicable in any given language. The former class of types are stored in a core-grammar file which is output with every Matrix-derived grammar and the latter class of types are implemented in "libraries" which allow interactive customization based on the answers given in the questionnaire. Each subpage of the questionnaire corresponds generally to a library of the system. However, choices across libraries often interact so the dimensions of variation in output grammars are complex.

2.1.1 System Architecture

The overall architecture of the Grammar Matrix customization system is shown in Figure 2.1. The user's choices on the questionnaire are stored in a choices file. It is known that certain sets of choices create dependencies on other choices and other sets of choices are incompatible with each other. So, as the user completes the questionnaire, a system of validation provides feedback about what has been selected so far and what is still required. Validation is discussed further in 2.1.3, below. Once the user has a valid choices file, the customization script can be run to create a grammar.

A validated choices file serves as input to the customization script. The customization script fetches the core-grammar containing types and constraints which are posited to be useful for all natural languages and includes them in the output. Then the script provides the choices file to a series of subscripts which generally correspond to the libraries of the system. The libraries compute the definitions for customized types based on the choices and these are included in the output grammar files. When the customization script is complete, the grammar is packaged as an archive file which when decompressed is ready to be processed using standardized tools.

Figure 2.1: The overall architecture of the Grammar Matrix customization system (Bender et al., 2010b, 29)



2.1.2 Regression tests

In part because the library systems can interact with each other, creating a complex space of cross-linguistic variation, and because development of the Grammar Matrix is carried out by different people at different times, it became necessary to develop a system to ensure that future developments to the Grammar Matrix system do not cause previous work to malfunction. To address this, the Grammar Matrix includes a suite of tests which provide both a validation layer for implemented analyses and an insurance against "regression" as the system is further developed (Bender et al., 2007). These tests consist of a choices file, a test-suite of sentences, and a gold standard semantic representation for those sentences. Tests are created each time a developer adds some functionality to the customization system. If future development breaks some older functionality in such a way that the customization system no longer maps a test-suite to the gold standard seman-

tics (given a choices file), then a regression has occurred and either the new development must be reworked to respect the progress to date or the old system must be updated for compatibility with the newer work.

Part of the work of creating a library for the customization system is creating a set of tests related to that library which exhibit its functionality. I present the tests related to the negation library in Chapter 6.1.

2.1.3 Validation

Because the questionnaire provides a vast space of choices, and not all of them were specifically envisioned by the developers to lead to coherent grammars, a system of validation for questionnaire responses is also in place. The validation system runs each time a user submits data from a subpage of the questionnaire and provides feedback at three levels: errors, warnings, and infos. Below follow examples of each.

A choices file with validation errors is prevented from being input to the customization system. As an example, consider this question from the word-order subpage: "Does your language have auxiliary verbs? (yes) (no)". A user can answer this question in the negative and then go ahead and define auxiliary verbs on the lexicon subpage. Such a choices file has to be prevented from being submitted to the customization script because the auxiliary verbs defined on the lexicon page will not have appropriate supertypes (because of the answer provided on the word-order page). This is accomplished by placing a validation error on the errant choice with a piece of feedback for the user: "You have defined an auxiliary verb and also said your language does not contain auxiliary verbs." Once a user corrects any validation errors, the choices file can be submitted to the customization system.

There is a second class of validation results called warnings. These signal to the user that a particular set of choices may provide limited usability, undefined behavior, or alternatively, that providing an answer for a particular choice is highly recommended. For example, on the general information subpage, leaving the question about whether the Grammar Matrix developers can archive a user's choices unanswered results in a valida-

tion warning and provides the following feedback to users: “Please answer whether you will allow your answers to be retained.” Choices files with validation warnings can be input to the customization system.

The final class of validation results is termed infos and provides further feedback about a choice or set of choices, but without any negative connotation. As an example, in defining type hierarchies for lexical types on the lexicon page a user can define a feature on a supertype which is inherited by subtypes. After saving the page, the customization system prints infos on the subtypes which display the matrix of inherited features. A second example of infos is employed on the general information page. When a user enters an ISO-639-3[2] code for the language being described, the provided code is looked up in the code table and if found, an info is printed which suggest the proper reference name for the language to the user.

In creating a negation library for the Grammar Matrix, I will also employ the validation system to rule out combinations of choices which are known not to lead to coherent grammars, to warn users about potential downstream details they will need to consider given their choices and to print infos for the user when non-negative live feedback about their choices can be computed.

2.1.4 Goals of the thesis

The existing library on sentential negation for the Grammar Matrix is one of the earliest additions to the system. Because early development work was concentrated on getting the system up and running, this library does not have the foundations in typological research that other, later, libraries do have. Therefore, this thesis proposes a new sentential negation library for the LinGO Grammar Matrix. That will entail the following:

1. a review of the linguistic (typological and morphosyntactic) literature to discover common morphosyntactic realization strategies for marking sentential negation and insights as to their formal analysis

[2]These three letter language identification codes are also shown in brackets on examples throughout this thesis document, `http://www.sil.org/iso639-3/`.

2. the development of a set of HPSG analyses integrated with the Grammar Matrix's core-grammar and library framework

3. a set of options for user-linguists and a mapping from those choices to the analyses developed

4. a set of tests, each with three parts (choices, test-suites, gold standard) which ensure both the validity of the analysis and guard against future regression

These points are addressed beginning with the review of literature in Chapter 3. Chapter 4 details the proposed analyses and their integration with the Grammar Matrix types. Chapter 5 describes the architecture of the questionnaire and the choices made available to users. In Chapter 6, I present the regression tests developed for the library described here. In the rest of this chapter, I describe some general properties of HPSG which are important for the exploration in this thesis.

2.2 HPSG and typed feature structure grammars

In HPSG, all grammatical objects are modelled as typed feature structures. Lexemes, words, phrase-structure rules, lexical rules are all seen as complex feature structures arranged into a type hierarchy. Grammatical description takes place as the elaboration and development of this hierarchy. The type hierarchy defines the ontology of possible structures. Well-formed sentences are those which unify (or are compatible with) a so-called initial symbol.[3] This initial symbol is defined as a set of constraints upon a grammatical type defined within the hierarchy.

Feature structures are usually presented as attribute-value matrices (AVM), (1) provides some examples. (1a) shows some constraints on a lexical entry for a verb like *love* (adapted from Kim and Sag 2002). (1a) also shows the recursive nature of the typed feature structure ontology. The value of the feature HEAD is itself a feature structure (of type

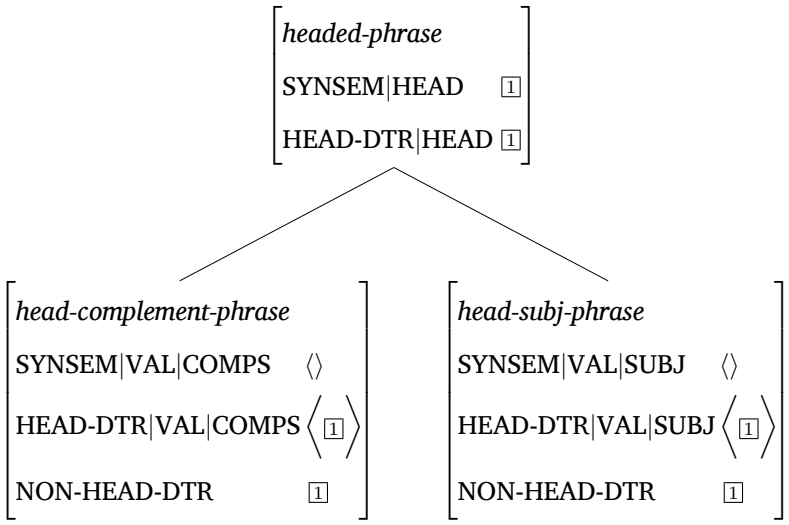
[3]The term "initial symbol" is taken from formal language theory whereby several classes of grammars are defined as a n-tuples which include a set of production rules P, and S, a set of initial symbols to which productions apply (cf. Hopcroft et al. 2006). For linguists modelling natural language, this initial symbol is usually a formalization of the notion "sentence", or "matrix clause".

verb). (1b) shows some of the constraints on a feature structure model of a phrase headed by a finite verb. One aspect of the typed feature structure framework is the identity constraint. This constraint can be defined to hold between multiple feature paths and ensures not just type but token identity on those nodes (in graph terminology this property is called reentrancy). In example (1b), the value of the HEAD feature is identical to the value of the HEAD feature of the head daughter of the construction (in effect, implementing the head-feature principle (Pollard and Sag, 1994)). The example in (2) shows how arranging types into an inheritance hierarchy allows for a constraint like the head-feature principle to be stated only once. The type *headed-phrase* is defined generally and phrases which require the constraints it introduces are its daughter types with further specifications.

(1) a.
$$\begin{bmatrix} \textit{v-lxm} & \\ \text{PHON} & \langle \textit{love} \rangle \\ \text{SYNSEM} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{verb} & \\ \text{AUX} & - \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SUBJ} & \langle \text{NP} \rangle \\ \text{COMPS} & \langle \text{NP}[\textit{acc}] \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

b.
$$\begin{bmatrix} \textit{headed-phrase} & \\ \text{SYNSEM} & \begin{bmatrix} \text{HEAD} & \boxed{1} \begin{bmatrix} \textit{verb} & \\ \text{FORM} & \textit{fin} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SUBJ} & \langle\rangle \\ \text{COMPS} & \langle\rangle \end{bmatrix} \end{bmatrix} \\ \text{HEAD-DTR} & \begin{bmatrix} \text{HEAD} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

(2)



In developing feature structure analyses for the various negation types, this work will be highly dependent on the existing type-hierarchy provided by the Matrix core-grammar as well as the types provided by the libraries. Mostly, this involves adding subtypes of existing types in the system and/or adding constraints to existing types. To aid exposition of these types, I will at times show proposed type definitions as Type Description Language statements (TDL) (Copestake, 2002) as opposed to the traditional AVMs shown in (1). TDL is the machine-readable language used to define types in DELPH-IN grammars. Unlike the traditional HPSG feature structure displays, TDL statements directly encode inheritance from supertypes (rather than relying on a separate statement of the type hierarchy), as shown in (3). A TDL version of the type definition shown in (2) for *head-complement-phrase*.

(3)
```
head-complement-phrase := headed-phrase &
    [ SYNSEM.VAL.COMPS < >,
      HEAD-DTR.VAL.COMPS < #comp >,
      NON-HEAD-DTR #comp ].
```

Next, I provide a few more details about lexical integrity in the Grammar Matrix, then I briefly describe the semantic representations and methods for semantic composition assumed here.

2.2.1 Lexical Integrity and Morphotactics

Bresnan and Mchombo (1995) argue that the grammatical system used in word formation

is distinct from the rules which combine words into phrases. This position is called the Lexical Integrity Hypothesis and it plays a large role in the architecture of the grammars output by the customization system.

This hypothesis is implemented by the LKB framework for grammar development (Copestake, 2002) and other DELPH-IN processing systems. The LKB allows grammar writers to create lexical rule instances and syntactic rule instances (stored in separate computer files) and will enforce that lexical rules can only apply before (i.e. lower in the tree than) any syntactic rules have applied. Additionally, the morphotactics library of the Grammar Matrix introduces a constraint on the daughters of phrase structure rules: they must be able to unify with [INFLECTED *infl-satisfied*] (Goodman and Bender, 2010). This constraint is used in Matrix-derived grammars to allow developers to ensure that required lexical rules do apply to lexical types before they can be daughters of phrase structure rules.

Goodman and Bender (2010) posit that the definition of *infl-satisfied* can vary by word classes of varying generality with further information contributed by a system of lexical rules and their co-occurrence relations. Goodman and Bender term this a problem of morphotactics. In their system a word form is only *infl-satisifed* if it has undergone all (transitively) required lexical rules. Lexical rules are organized according to position classes which facilitate the description of complementary distribution. Note that lexical rules do not have to contribute phonological changes so null affixation is possible. Finally, for my purposes, it is important to note that one lexical rule can require (or forbid) the application of another. This morphotactic infrastructure provides much of the groundwork that my analyses of inflectional negation will rely upon.

In this subsection I have pointed out the theoretical claim that word-building rules (morphology) are distinct from phrase-building rules (syntax) and that the former "precede" the latter. This constraint is built into the architecture of the grammars output by the Grammar Matrix system. Also, as will be revealed in the exposition of negation types presented from Chapter 3 and forward, negation can be expressed via lexical rules which may be required (or which may place a co-occurrence requirement on the presence of another lexical rule, forcing the second to be required). Therefore I have reviewed above the constraint on the value of the feature INFLECTED, which ensures all required lexical rules

have applied. Another aspect of the morphotactic infrastructure of the Grammar Matrix is that lexical rules are defined to stand in *position classes*, which allows complementary distribution of morphemes that fall into the same position class. A bound morpheme is modelled by a lexical rule which may or may not contribute phonological information (null affixation is possible). Finally, I reviewed the fact that lexical rules can place co-occurrence relations on other lexical rules enabling rule interactions (one rule can require or forbid the occurrence of another). These facts set out the available framework for defining syntactic dependencies between bound morphs as I move forward to model sentential negation constructions in future chapters.

2.2.2 Semantic composition and MRS

In this section I will review the basics of a representation scheme for semantics currently used in Matrix-derived grammars: Minimal Recursion Semantics (MRS) (Copestake et al., 2005). After exemplifying the basics, I will show how negation is represented in this system.

This thesis adopts the hypothesis that syntax constrains semantic structure, but only partially. Assuming a surface syntax approach, examples such as (4) are enough to show that the mapping from a syntactic structure to a semantic one is not just one-to-one. MRS allows underspecification in certain crucial ways, as explained below. For any such underspecified MRS, it can be fleshed out into a set of fully specified logical representations.[4] An intermediate representation of this sort is extremely valuable in implemented systems. Grammar writers can specify constraints of syntax on semantics, without having to compute a full set of semantic representations.[5]

In more detail, MRS representation is a "flat" semantic representation, where *Elementary Predications* (EPs) are collected in a bag.[6] The MRS structure also contains a list of constraints on scope taking properties of these EPs. In this way, an MRS can disallow cer-

[4]The procedure for generating this set is described in Copestake et al. 2005.

[5]In fact, many NLP tasks do not need fully specified logical forms and often do not want them.

[6]Even though a list structure provides the implementation for the bag, EPs in an MRS are not ordered.

tain scopal configurations but allow others to interact, as described in more detail below.

The fundamental insight is that a semantic representation for (4a) needs to list all the predications involved and to correctly specify which entities are arguments of which EP, but should be compatible with both scopal readings shown in (4b) and (4c).

(4) a. Every athlete has some goal.

b. $\forall x \exists y [athlete(x), goal(y), has(x, y)]$

c. $\exists y \forall x [athlete(x), goal(y), has(x, y)]$

To achieve this, MRS proposes that there are (at least) two types of elementary predications: those which require a scope demarcation for interpretation and those which do not. The former class includes quantifiers, scopal (non-intersective) modifiers, and any head with a clausal argument. The latter includes, for example, most other lexical verbs and common nouns.

MRS lists all the predicates of a sentence in a flat list (and gives each one a "handle" or identifier which allows it to be referenced). The predications which do not require scope demarcation will have argument slots corresponding to individual or event type variables, those requiring scope demarcation will have argument slots corresponding to their scope (so-called "holes"). In our example above *every* and *some* will have one basic-type argument slot for the bound variable and two scopal holes for the restriction and body of the quantifier. Holes are also given handles, which might correspond to the label of another predicate. When the handle in a hole doesn't correspond to the label of a predicate, the scope it indicates is underspecified. In (5a), I begin to build up an MRS representation, *h1* is the label for *every*, x is the bound variable of *every*, and *h2* and *h3* are the restriction and body of every (respectively). Also in (5a), *h2* is also the label of an EP (*athlete*), so *athlete* is in the body of *every*.

(5) a. EPS: $h1 : every(x, h2, h3), h2 : athlete(x)$

b. EPS: $h1 : every(x, h2, h3), h4 : athlete(x)$

But in (5b), both the restriction and body of *every* are underspecified. The second aspect of the system is that certain scopal readings are constrained by syntax. In our example under discussion, we know that *athlete* should appear in the restriction of *every*, and *goal* should appear in the restriction of *some*. To achieve this constraint an MRS also includes a list of constraints on the scope configurations by specifying relationships between handles. Every scope hole has a handle and every EP has a handle. So, HCONS (or handle constraints) can list a special sort of equivalency termed qeq (also written $=_q$) for "equal modulo quantifiers". A qeq equates two handles while allowing the possibility that a quantifier intervenes.

In (6), I present a partial MRS for the example where each EP and hole has its own handle, and HCONS are introduced to restrict interpretations.

(6) EPS: $\{h1 : every(x1, h2, h3),$
$h4 : athlete(x1),$
$h5 : has(x1, x2),$
$h6 : some(x2, h7, h8),$
$h9 : goal(x2)\}$
HCONS: $\{h2 =_q h4, h7 =_q h9\}$

The partial MRS in (6) shows a semantic representation for (4a) which is compatible with both readings (4b) and (4c). To see how, note that the second argument to `every`, *h2*, is $=_q$ with *h4* and *h4* corresponds to *athlete*. Likewise, *goal* is constrained to appear within the restriction of *some*. However, the body of both quantifiers is left unconstrained—giving rise to interpretations with both wide and narrow scope for *every*.[7]

MRSs also have two other handles, a global and local top handle (GTOP and LTOP) which allow full MRSs to reference each other in scope configurations. It's also important to note that MRSs are straightforwardly encoded in a typed feature structure: (7) is a full MRS for the example under discussion in AVM notation.

[7]There are further constraints on the well-formed MRSs which affect interpretation but which are not discussed here for reasons of space (Copestake et al., 2005).

(7)
$$\begin{bmatrix} \textit{mrs} \\ \text{GTOP} \quad \textit{h0} \\ \text{LTOP} \quad \textit{h0} \\ \text{RELS} \quad \left\langle \begin{bmatrix} \textit{arg12ev-rel} \\ \text{PRED} \quad \text{“\_has\_v\_rel”} \\ \text{HANDLE} \quad \textit{h4} \\ \text{ARG1} \quad \textit{x1} \\ \text{ARG2} \quad \textit{x2} \end{bmatrix}, \begin{bmatrix} \textit{arg1ev-rel} \\ \text{PRED} \quad \text{“\_athlete\_n\_rel”} \\ \text{HANDLE} \quad \textit{h4} \\ \text{ARG1} \quad \textit{x1} \end{bmatrix}, \begin{bmatrix} \textit{arg1ev-rel} \\ \text{PRED} \quad \text{“\_goal\_n\_rel”} \\ \text{HANDLE} \quad \textit{h9} \\ \text{ARG1} \quad \textit{x2} \end{bmatrix}, \right. \\ \left. \begin{bmatrix} \textit{scopal\_rel} \\ \text{PRED} \quad \text{“every\_rel”} \\ \text{HANDLE} \quad \textit{h1} \\ \text{BV} \quad \textit{x1} \\ \text{RESTR} \quad \textit{h2} \\ \text{BODY} \quad \textit{h3} \end{bmatrix}, \begin{bmatrix} \textit{scopal\_rel} \\ \text{PRED} \quad \text{“some\_rel”} \\ \text{HANDLE} \quad \textit{h6} \\ \text{BV} \quad \textit{x2} \\ \text{RESTR} \quad \textit{h7} \\ \text{BODY} \quad \textit{h8} \end{bmatrix} \right\rangle \\ \text{HCONS} \quad \left\langle \begin{bmatrix} \textit{qeq} \\ \text{HARG} \quad \textit{h2} \\ \text{LARG} \quad \textit{h4} \end{bmatrix}, \begin{bmatrix} \textit{qeq} \\ \text{HARG} \quad \textit{h6} \\ \text{LARG} \quad \textit{h9} \end{bmatrix} \right\rangle \end{bmatrix}$$

In this sort of system, monotonic semantic composition is straightforward. In a syntactic phrase, all daughters pass their semantics up to the phrasal node dominating them. Their lists of elementary predications are concatenated into a larger list on the mother nodes, and their lists of handle constraints are similarly combined. Note that rules can also contribute semantics. The feature C-CONT (mnemonic for construction content) records semantics contributed by a lexical or phrasal rule.

How is sentential negation to be encoded negation in an MRS? In this thesis negation is treated as a grammatical *Elementary Predication* which has a single scopal hole. I treat negation's argument as a scopal hole because quantifiers can intervene. In (8), if negation were treated as non-scopal, there would be no way to get the reading in which negation outscopes *every* because there would be no handle (no hole) for a scopal position under negation (Emily Bender, p.c., Flickinger 2000). Constraints on lexical and phrase structure rules will constrain the scope of negation's scopal position according to the method of grammatical attachment, placing the. $=_q$ constraint between the handle of negation's scopal hole and the handle of the main verb's key predication.

(8) a. Kim didn't read every book.

b. $every(x, book(x), neg(read(K, x)))$

c. $neg(every(x, book(x), read(K, x)))$

(9)
$$\begin{bmatrix}
\text{GTOP} & h0 \\
\text{LTOP} & h1 \\
\text{RELS} & \left\langle \begin{bmatrix} \textit{proper_q_rel} \\ \text{HANDLE} & h3 \\ \text{BV} & x5 \\ \text{RESTR} & h4 \\ \text{BODY} & h6 \end{bmatrix}, \begin{bmatrix} \textit{named-rel} \\ \text{NAME} & \text{“Kim”} \\ \text{INDEX} & x5 \\ \text{HANDLE} & h7 \end{bmatrix}, \begin{bmatrix} \textit{event-rel} \\ \text{PRED} & \text{“neg_rel”} \\ \text{INDEX} & e10 \\ \text{HANDLE} & h8 \\ \text{ARG1} & h9 \end{bmatrix}, \begin{bmatrix} \textit{event-rel} \\ \text{PRED} & \text{“_read_v_rel”} \\ \text{INDEX} & e2 \\ \text{HANDLE} & h11 \\ \text{ARG1} & x5 \\ \text{ARG2} & x12 \end{bmatrix}, \right. \\
& \left. \begin{bmatrix} \textit{scopal_rel} \\ \text{PRED} & \text{“every_rel”} \\ \text{HANDLE} & h13 \\ \text{BV} & x12 \\ \text{RESTR} & h14 \\ \text{BODY} & h15 \end{bmatrix}, \begin{bmatrix} \textit{individual_rel} \\ \text{PRED} & \text{“_book_n_rel”} \\ \text{INDEX} & x12 \\ \text{HANDLE} & h16 \end{bmatrix} \right\rangle \\
\text{HCONS} & \left\langle \begin{bmatrix} \textit{qeq} \\ \text{HARG} & h4 \\ \text{LARG} & h7 \end{bmatrix}, \begin{bmatrix} \textit{qeq} \\ \text{HARG} & h9 \\ \text{LARG} & h11 \end{bmatrix}, \begin{bmatrix} \textit{qeq} \\ \text{HARG} & h14 \\ \text{LARG} & h16 \end{bmatrix} \right\rangle
\end{bmatrix}$$

Therefore, in this thesis, I will be seeking to create syntactic types which introduce a `neg_rel` with a scopal argument position, and a $=_q$ (qeq) relation specified on the HCONS between the scopal argument position and the handle of the MRS for the syntactic element it combines with. Negative verbs will qeq their complement's predication's handle, negative modifiers will qeq their head's predication's handle, negative inflectors will do the same to their stem's predication's handle. The question of how different attachment mechanisms (modification, complementation, inflection) affect the distribution of negative scope is not explored in this thesis. But it is noted that fixing the scope of negation with respect to its attachment point does bear predictions for the interpretation of negation. To illustrate with a full example, (9) provides an MRS representation for (8) (shown here as an

AVM adapted from the online demonstrator of the English Resource Grammar (Flickinger, 2000).[8] The structure contains EPs corresponding to the semantic contribution of each lexical item in the sentence along with one quantifier relation (*proper_q_rel*) contributed by the phrase structure rule which licenses proper names as NPs. The representation also shows the HCONS which bear upon enumerating fully scope resolved interpretations. This MRS constrains scope in such a way as to allow both the readings shown in (8b) and (8c), as discussed above.

This concludes the background information regarding MRS and semantic composition with negation. In the next chapter, I begin to build up an inventory of negation types by surveying grammatical and typological information.

[8] `http://erg.delph-in.net/logon`

Chapter 3

REVIEW OF NEGATION LITERATURE

In this chapter,[1] I provide a short tour through some of the relevant linguistic literature on negation. This survey is intended to highlight ideas and concepts directly relevant to the typology and associated analyses presented in this thesis. For a more comprehensive bibliography on negation, see chapter 10 of Horn 2010. In the first half of this chapter, I review the properties of grammatical morphemes as described in HPSG literature. Then, in the second part, I examine typological surveys of negation constructions. In the final section, I preview the integration of these works into a new typology of syntactic constructions. Chapter 4 then takes up this typology in more detail.

3.1 Morphemes in HPSG

Dryer (2005, p. 454) found that "all the ways of indicating negation include negative morphemes". Following Dryer, this work assumes that negation must be indicated in a sentence by some lexical material, and that lexical material is composed of morphemes. Therefore, the first question to ask regards the relevant dimensions of variation for morphemes types.

Given the Lexical Integrity Hypothesis (Bresnan and Mchombo, 1995) discussed in the last chapter, word-internal structure is generated by a different set of rules (or constraints) from those that combine words into phrases. So, in creating a typology for sentential negation, the first property to be distinguished is the bound or free status of the negator.

Opposed to morphological material which is necessarily bound, fully inflected word forms (stems after any required affixation) may enter into a syntactic phrase as either a head or a dependent (Zwicky, 1985, Sec. 2). As we will see from the typological evidence, both head and dependent negation markers have been reported.

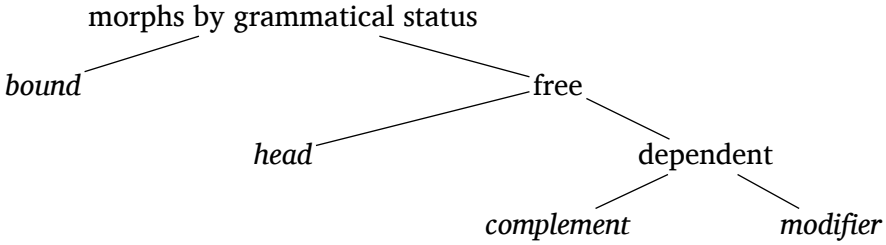
When a morpheme is a dependent, HPSG theory (as X-bar theory, before it) contains

[1] Portions of this chapter and the next were published as Crowgey (2012a).

a distinction between arguments and modifiers. The tradition in many branches of syntax has been to treat free negators which are syntactic dependents as modifiers: they occur optionally and can show up in adverb positions (Pollock, 1989). However, HPSG theorists (Kim and Sag, 2002) have argued for the treatment of negators of finite verbs in French and English as selected complements.

Given the considerations above, we can create a partial typology of grammatical morphemes in HPSG as in (10). Below, these properties will be integrated into a broader typology of predicted negation types.

(10)



3.2 Typological surveys

Östen Dahl's (1979) survey of 240 languages offers a typology with 5 sentential negation categories. In (11) I list his categories and where possible I give Dahl's example of each.

(11) a. morphological negation[2]

b. **uninflected negation particles**

English

John is not swimming. [eng]

c. **negative auxiliary**

Finnish

[2]Dahls's article presents a careful discussion of the difficulty of determining morphological from syntactic attachment when looking at language data, but he does not provide any direct examples from the languages he classified as marking negation through morphology. Instead, the work presents edge cases which exemplify the difficulty of determining a strict criteria for what counts as morphology. The article includes an appendix with a table showing the classification and readers are referred to the original grammars to find primary data.

En lue
NEG.1SG_SUBJ read.NONFINITE
I do not read. [fin]

d. **dummy auxiliary construction**
English
John smokes → John does not smoke. [eng]

e. **double particle construction**
French
Je ne sais pas. [fra]

In discussing his results, Dahl relates his work to questions of universal tendencies in the placement of negation, hearkening back to Jespersen's (1924) discussion of diachronic changes typically undergone by negation constructions. However, for my work, I am interested in the synchronic syntactic constructions which underly Dahl's categories and these are not addressed directly in his discussion. In going forward with these categories, I am to some degree forced to interpret them syntactically from the standpoint of HPSG, which, as I show below, provides natural distinctions which line up neatly with Dahl's typological categories.

As discussed above in Chapter 2.2.1, Grammar Matrix output grammars implement a distinction between morphological and syntactic attachment, so Dahl's morphological negation as seen an analog to the *bound* node of (10). Viewed from an HPSG perspective, then, this sort of construction can be modelled as sentential negation marked by an inflectional marker on the verb. Examining Dahl's category *negative auxiliary* from the perspective of HPSG reveals that these correspond to syntactic heads (10), that is, negative auxiliary verbs. Dahl's category of *uninflected negation particles* may be taken to indicate free morphemes which negate, in terms of HPSG grammatical types, these correspond to the *dependent* node of (10).

The next category in Dahl's typology, *dummy auxiliary construction*, is spurious given my task. Dahl is discussing a construction as found in English, where a negator is dependent on

an auxiliary verb head which must be present in a negated sentence. While this category is certainly helpful in Dahl's investigation of negative placement, for my purposes it is subsumed by systems already present in a grammar. For example, a grammar of English will already implement a system of finiteness constraints and lexical selection. If the *not* of sentential negation is selected for by the dummy auxiliary and not by finite lexical verbs, then the dummy's presence is a side effect of the fact that grammatical complements have selecting heads.[3] For my purpose here, the case of English negation is already captured by a full description of the properties of the grammatical dependent *not* within the context of an implemented English grammar.

Another important survey of negators is Matthew Dryer's (2005) survey in WALS. He employs a set of categories which is very similar to Dahl's. Dryer's classification is shown in (12).

(12) a. **negative affix**

Kolyma Yukaghir (Yukaghir; Russia; Maslova 2003, 492; Dryer 2005)

met numö-ge el-jaqa-te-je

1SG house-LOC NEG-achieve-FUT-INTR.1SG

I will not reach the house. [yux]

b. **negative particle**

Musgu (Chadic; Cameroon, Chad; Meyer-Bahlburg 1972, 186; Dryer 2005)

à sədə̀ cécébè pày

3SG.M know jackal NEG

He didn't see the jackal. [mug]

c. **negative auxiliary verb**

Finnish (Uralic; Finland; Sulkala and Karjalainen 1992, 115; Dryer 2005)

e-n syö-nyt omena-a

NEG-1SG eat-PTCP apple-PART

I didn't eat an apple. [fin]

---

[3]Cf. Sag et al. 2003 for details of a basic HPSG analysis of do-support.

d. **negative word, unclear if verb or particle**

Maori (Austronesian; New Zealand; Bauer 1993, 140; Dryer 2005)

kaahore taatou e haere ana aapoopoo

NEG 1PL.INCL T/A move T/A tomorrow

We are not going tomorrow. [mri]

e. **double negation**

Ma (Atlantic-Congo; DRC; Tucker and Bryan 1966, 130; Dryer 2005)

tá-mù-sùbù-li nɔ́ŋgbɔ́ nyɔ̀

NEG-1SG-eat-PST meat NEG.1SG

I did not eat meat. [msj]

Dryer's categories line up with Dahl's in several instances. (13a) and (13c) are clearly corollary to (12a) and (12c). Unparalleled in Dahl is Dryer's category for data which is unclear as to syntactic type, verb or particle. Viewing Dryer's categories from the viewpoint of HPSG, it seems reasonable to treat Dryer's negative particles, as I treat Dahl's, as corresponding to the *dependent* node of (10)—word forms which are either arguments or modifiers. Unlike Dahl, Dryer does not include the dummy auxiliary construction as a separate category, but like Dahl, Dryer leaves the double negation category undifferentiated.

(13)

HPSG	Dahl	Dryer
bound	morphological neg	neg affix
head	neg auxiliary	neg auxiliary
dependent	neg particle	neg particle
	dummy auxiliary	
		neg word, unclear
	double particle	double negation

The table in (13) shows the correspondences we have seen so far between grammatical theory and typological categories of sentential negation from Dahl and Dryer. The typological work and its correspondence to HPSG theory motivate the claim that sentential negators can be drawn from any one of at least three syntactic categories: affixes, heads

and dependents. But in the typology of HPSG lexical material offered above, dependents are analyzable into two further subcategories: arguments and modifiers. So we have to ask whether sentential negation can be realized as either or both of these subtypes. The answer to this question can be found by consulting the HPSG for analyses of negation in particular languages. In fact, as mentioned above, Kim and Sag 2002 argue that English *not* of finite (sentential) negation is in fact a grammatical complement of the auxiliary that obligatorily heads the sentence, while the *not* of non-finite (VP) negation is attached by a head modifier rule. So both subtypes of *dependent* are attested in the HPSG literature on negation.

In drawing together the analysis of Kim and Sag along with the typological surveys of Dahl and Dryer, I find that all four types of grammatical morphemes shown in (11) are attested as sentential negators in simple (single) negation constructions.

The next step is to look further at the category of double negation. In Dryer's survey of 1159 languages, 120 are coded as double negation. This is a significant minority of the survey. Furthermore, Dryer codes languages like French, which display variation between single and double negation, as one of the single negation types. So if anything, the importance of double negation constructions is underrepresented in Dryer's results and a model of sentential negation for the Grammar Matrix needs to deal with double negation phenomena competently.

From the perspective of implemented HPSG, the undifferentiated category of "double negation" is too vague. As made clear by the discussion above, each lexical item in HPSG is defined rigorously in terms of grammatical properties. The proposal to create a double negation construction in a Matrix-style grammar immediately brings up further questions about how each of the two exponents of negation are grammatically attached and how (by what feature constraints) mutual dependency is engineered. The model I propose in this thesis builds on the typology of lexical material shown in (11), to capture simple and double negation. The model is derived by applying the idea that each of the two lexical items in a double negation, just like the single lexical item in a simple negation construction, needs to be defined to belong to one of the grammatical categories in (11). Thus the notion of exponence is promoted to an independent parameter of the model which cross factors

with morpheme type. The idea being that a single negation construction can draw one morpheme type from the morphemic typology, a double negation construction can draw twice.[4] Figure 3.1 shows the typology of negation types proposed in this thesis.

In the next chapter, I step through each of the negation types in Figure 3.1 and present the details of each analysis.

[4]I note here that this methodology also predicts the possibility of negation constructions that are tripartite, quatripartite, etc. In fact, Budd (2010) proposes that Bierebo [bnk] exhibits tripartite negation. Because of its typological rarity, tripartite negation is not treated in this thesis, but is set aside for examination in future work.

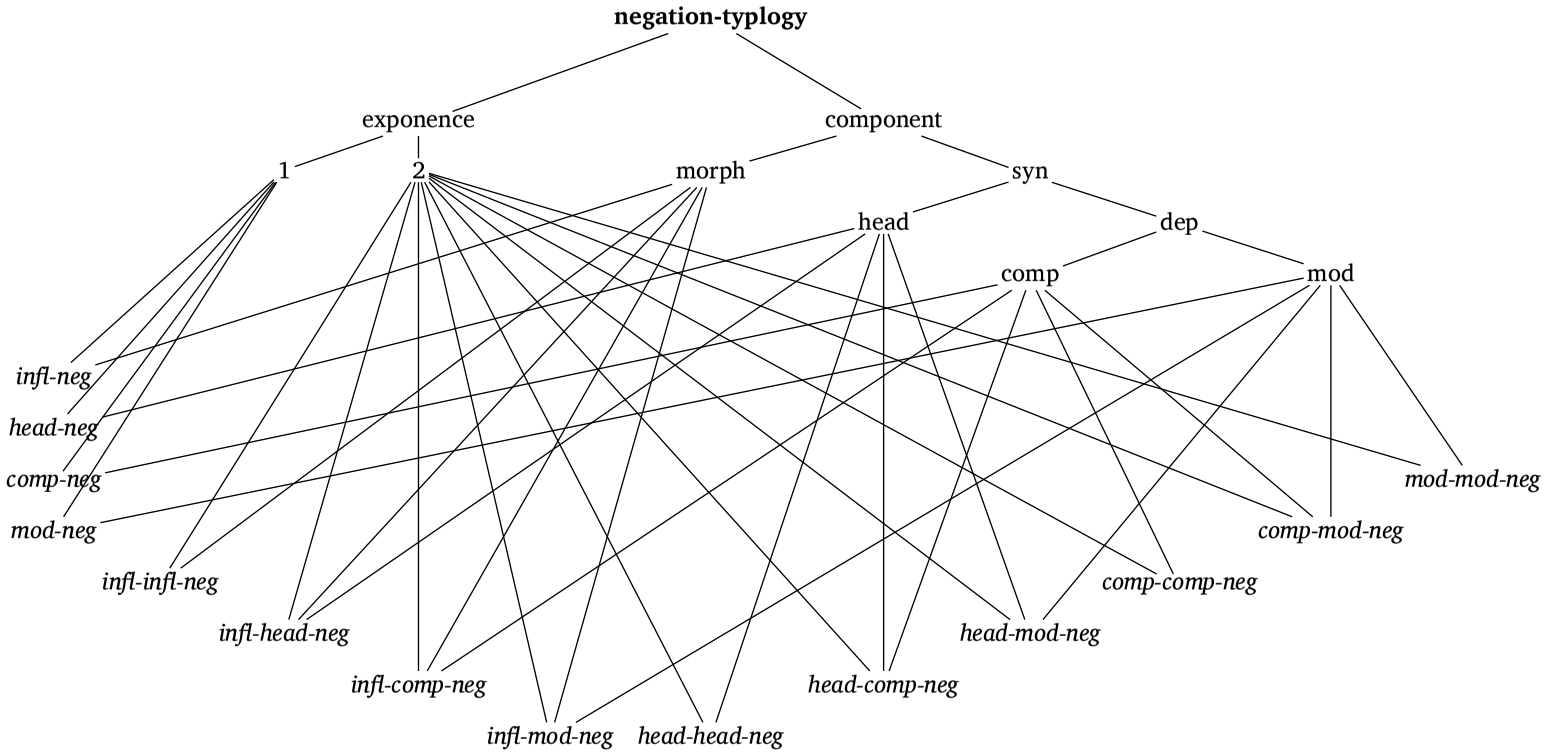


Figure 3.1: Proposed negation typology with exponence crossfactored against grammatical morpheme types

Chapter 4

ANALYSIS

In this chapter I provide more details about the model for sentential negation strategies proposed in Chapter 3. Specifically, I sketch HPSG analyses for each of the negation types in the typology. I will first step through the simple negation types and show feature structures and example languages (where possible). Then, I turn to the details of the bipartite (or double) negation types.

4.1 Simple negation types

As mentioned in the last chapter, simple negation types attested by typological surveys have been predicted by the theory of lexical material in HPSG. In this section, I present four examples from natural language data which may fit the four simple negation types of the model I propose as well as feature structure descriptions of those types.[1] The four are negation by inflection, by auxiliary verb, by complement or by modifier.

4.1.1 Negation by inflection

(14) is an example from Achumawi [acv] (Dryer, 2005; De Angulo and Freeland, 1930) of a bound morphological negator which attaches to an auxiliary verb.[2] This type of morpheme can be modelled straightforwardly as an inflectional rule which attaches to auxiliary verbs and contributes the negation relation through C-CONT. An AVM representation of such an inflectional rule is shown in (15). As discussed above, the feature INFLECTED is used by the Grammar Matrix's system of morphotactics (Goodman and Bender, 2010).

[1]The AVMs below show feature-structures of grammatical types output by the library. Discussion of these types from the perspective of their assembly in a grammar is presented in Chapter 5.2.

[2]Parallel to the English construction, here we see a dummy auxiliary introduced as the host to the negator, but the auxiliary is not itself a negative word.

(14) a. s-ä˙m-á
1SG-eat-FV
I eat. [acv]

b. tsé-s-ùw-í d-ámm-ì
NEG-1SG-be-FV NMLZ-eat-FV
I do not eat. [acv]

(15)
$$\left[\begin{array}{ll}
\textit{negpc-lex-rule} & \\
\text{INFLECTED}|\text{NEGPC-FLAG} & + \\
\text{C-CONT} & \left[\begin{array}{ll}
\text{HOOK} & \left[\begin{array}{ll} \text{XARG} & \boxed{1} \\ \text{LTOP} & \boxed{2} \\ \text{INDEX} & \boxed{3} \end{array}\right] \\
\text{RELS} & \left\langle !\left[\begin{array}{ll} \textit{event-relation} & \\ \text{PRED} & \text{"neg_rel"} \\ \text{LBL} & \boxed{2} \\ \text{ARG1} & \boxed{4} \end{array}\right] !\right\rangle \\
\text{HCONS} & \left\langle !\left[\begin{array}{ll} \textit{qeq} & \\ \text{HARG} & \boxed{4} \\ \text{LARG} & \boxed{5} \end{array}\right] !\right\rangle
\end{array}\right] \\
\text{DTR} & \left[\begin{array}{ll}
\textit{aux-verb-lex} & \\
\text{CONT}|\text{HOOK} & \left[\begin{array}{ll} \text{XARG} & \boxed{1} \\ \text{LTOP} & \boxed{5} \\ \text{INDEX} & \boxed{3} \end{array}\right]
\end{array}\right]
\end{array}\right]$$

4.1.2 Negation by auxiliary verb

(16) provides an example of a negator as a syntactic head in Finnish [fin], (Sulkala and Karjalainen, 1992, 115), (Dryer, 2005)—in this case an auxiliary verb which takes the lexical verb to be negated as a complement. This negative auxiliary verb can be modelled as contributing the negation relation through normal semantic composition of its own CONT value with that of its argument(s) via a head-complement rule. The Grammar Matrix type hierarchy provides analyses for semantically contentful auxiliaries as part of the approach

to major constituent word order (Fokkens, 2010), so the idiosyncratic properties of the negative auxiliary are rather minimal. We must specify the spelling and the name of the predicate, as in (17).

(16) e-n syö-nyt omena-a
NEG-1SG eat-PTCP apple-PART
I didn't eat an apple. [fin]

(17)
$$\begin{bmatrix} \textit{neg-aux-lex} & \\ \text{STEM} & \left\langle \text{“neg”} \right\rangle, \\ \text{SYNSEM} & \begin{bmatrix} \text{CONT|RELS} & \left\langle \begin{bmatrix} \textit{event-rel} \\ \text{PRED} \quad \texttt{neg\_rel} \end{bmatrix} \right\rangle, \\ \text{CAT|VAL|COMPS} & \left\langle \begin{bmatrix} \text{FORM} \quad \textit{nonfinite} \end{bmatrix} \right\rangle \end{bmatrix} \end{bmatrix}$$

4.1.3 Negation by selected complement and lexical rule

(18) shows a negated sentence of English [eng]. As mentioned above, (Kim, 2000; Kim and Sag, 2002) provide compelling arguments for treating the *not* of sentential negation as a selected complement of the auxiliary verb in the languages they analyze. For English, a complement-changing, non-inflecting lexical rule creates a version of the auxiliary which requires *not*, along with any other complements.[3] Some of the relevant properties of the COMPS-changing negation lexical rule are shown in (19). The lexical rule will apply to a finite auxiliary verb and add the negator's feature structure to the beginning of its COMPS list. The negative lexical item will add the semantic *neg_rel* via normal semantic composition in a head-complement phrase. The *qeq* between the negation relation and the lexical verb will be added to HCONS by normal constraints on head-complement phrases when the auxiliary combines with the lexical verb.

[3]As discussed in (Sag et al., 2003), this lexical rule treatment also sets up a parsimonious analysis of a family of syntactic phenomena for English auxiliaries, capturing the so-called NICE properties: Negation, Inversion, Contraction, Ellipsis.

(18) I do not care
1SG AUX NEG care
I do not care. [eng]

(19)
$$\begin{bmatrix} \textit{neg-lex-rule} & \\ \text{SYNSEM|COMPS} & \left\langle \left[\textit{neg-adv} \right] \right\rangle \oplus \boxed{1} \\ \text{DTR} & \left[\text{COMPS} \quad \boxed{1} \right] \end{bmatrix}$$

4.1.4 Negation by free modifier

(20) is an example from Ngas [anc] (Dryer, 2009; Burquest, 1973), which is perhaps best treated as a modifier for two reasons. The first has to do with linguistic tradition and recursion. Kim and Sag (2002) treat finite (sentential) negation as a complement of the auxiliary after arguments based on the specific facts of English and French. The ability to apply recursively is traditionally associated with modifiers and Kim and Sag show that (in English) non-finite (VP) negation can recurse, but finite negation cannot. From this we can take that recursion can be seen as a test for negation by modification.

Musa rok gik mwa duŋ-duŋ ka
NAME throw rock PL much NEG
Musa didn't throw many rocks. [anc]

Because this argumentation is language specific we cannot apply Kim and Sag's discussion directly, without more facts about the syntactic structure of Ngas. But the fact that negation by modification is argued for in at least some cases provides reason to include it in a model intended to be applicable for any language. The second reason to include negation by modification comes from the particulars of implementation. In the formalism of (Copestake, 2002), argument lists are implemented as plain linked-lists (cf. difference lists in the semantic representations) whose length is not generally known (at the level of generality required by a lexical rule for verbs).[4] Thus, a monolithic lexical rule engineered

[4]Linked lists support push and pop operations (akin to stacks). Placing an item on top of the stack is trivial. Finding the depth of the stack takes extra computation.

to insert an additional complement at the end of the argument list of any verb is not possible. Instead, a specific lexical rule will have to be written for each verb type according to COMPS list length. This approach leads to an unnecessary over-complication of the lexical rule system. But this complication is avoided if the negator is attached by a head-modifier rule. Because this work is intended to provide a family of analyses which should be useful for the widest possible range of languages and grammar writers. I have included negation by modification alongside negation by complementation. As always, grammar writers can use the customization system to create grammars to test which of the analyses is more appropriate for their language.

Negation by modification will require the introduction of a head-modifier rule and a negative modifier type to the grammar. An individual grammar can be customized so that the negative modifier combines with a VP (as in the Ngas example above) and occurs after its head. Some details of a definition for a modifier-negator such as this are shown in (20). The type of grammatical object modified, as well as placement before, after or on either side of the head are available as customizable options for this negation strategy (as detailed in Chapter 5).

(20)
$$\left[\begin{array}{ll} \textit{neg-adv-lex} & \\ \text{SYNSEM|CAT} & \left[\begin{array}{ll} \text{VAL} & \left[\begin{array}{ll} \text{SPR} & \langle\rangle \\ \text{COMPS} & \langle\rangle \\ \text{SUBJ} & \langle\rangle \end{array}\right] \\ \text{POSTHEAD} & + \\ \text{HEAD|MOD} & \left\langle \left[\begin{array}{ll} \text{LOCAL|CAT} & \left[\begin{array}{ll} \text{HEAD} & \text{verb} \\ \text{VAL} & \left[\begin{array}{ll} \text{SUBJ} & \textit{ne-list} \\ \text{COMPS} & \textit{empty-list} \end{array}\right] \end{array}\right] \end{array}\right] \right\rangle \end{array}\right] \end{array}\right]$$

The model presented here contains four types of simple negation which correspond to the four methods of grammatical attachment for lexical material discussed in Chapter 3. Now I turn to the bipartite types.

4.2 Bipartite negation types

There are 10 bipartite negation types predicted by the methodology described in the introduction. Here, I examine each of these types in more detail. There are two new considerations to take into account when constructing bipartite negation types: the first is the question of which element (if any[5]) is to bear negative semantics and which will be conceived of in terms of agreement; the second is a related question, once the bearer of negative force is determined, how will the dependency be engineered between the two elements?

The answer to the first question is to be determined based on the morphological status of the two negators in question. For example, because semantically empty affixes are less costly[6] than semantically empty free morphemes, the negative semantics would be placed on the free morpheme (when available). The answer to the second question is essentially determined on a case by case basis in a similar way—by considering the morphological status of the two negators in question. For example, when the bearer of negative force is an auxiliary and the secondary element is a free negator, the selectional properties of the auxiliary can be leveraged to require the second element. I give details of such considerations for each construction proposed below.

4.2.1 infl-infl-neg

Bipartite negation may be marked by two bound negators. Here, there are two subtypes to consider: (a) both negators are bound to the same head; (b) the negators are bound to separate heads. The case of (a) is attested, for example, in Izi-Ezaa-Ikwo-Mgbo [izi] (Dryer, 2009; Meier et al., 1975) (21) and in Spoken Egyptian Arabic [arz] (Lucas and Lash, 2008) (22).

---

[5]It is also possible to introduce negative force through a construction rule which can only occur given the presence of the two requisite morphemes (cf. Crysmann 2010). This approach is not pursued here.

[6]Costly, here, refers to expense of computational resource in using the grammar to generate strings from an MRS representation. Morphology can be modelled by finite state machines (Jurafsky et al., 2000), but syntax requires at least a context-sensitive model (Shieber, 1985), so the search space for required semantically empty words is much larger than that for bound forms.

(21) ó tó-òmé-dú ré
3SG NEG-do-NEG well
He does not do well. [izi]

(22) ma-bəḥibb-^{i}š migiyy-u hina ktīr
NEG-like.IMPF.1SG-NEG coming-his here much
I don't like his coming here a lot. [arz]

The (a) cases are readily modelled with existing approaches to implemented HPSG morphotactics, such as the one described in (Goodman and Bender, 2010) for the Grammar Matrix. As discussed in the introduction, one lexical rule can require the presence of another—and only one of the lexical rules will contribute the semantic relation and constraints shown in (17).

In the case of (b), with bound negators on separate heads, the only plausible situation is that one negator is bound to an auxiliary verb and the other to a lexical verb.[7] I have yet to find a report of such a construction, but the typology employed here predicts its existence. A schematic example of such a structure in a SVO language where auxiliaries precede their arguments (and raise the VP's subject) would look as in (23).

(23) np aux-neg1 iverb-neg2.

To create the dependency between the two elements, this sort of construction is readily captured through the selectional properties of the auxiliary and inflecting lexical rules. The lexical rule that attaches negation to the auxiliary will introduce negative semantics through C-CONT as in (18), but with the additional requirement that its lexical verb complement also be inflected for negation. To achieve this, the lexical rule will also constrain the COMPS value of the auxiliary to require a particular FORM value—one which the lexical rule attaching to the lexical verb will specify. (24) shows relevant parts of these lexical rules.

[7] If the putative second negator is bound to a nominal, it is best conceived of as a case of negative concord, a phenomenon distinct from bipartite negation, cf. de Swart and Sag 2002.

(24) a. $\begin{bmatrix} \textit{neg1-lex-rule} & \\ \text{SYNSEM|COMPS} & \left\langle \begin{bmatrix} \text{FORM} & \textit{negform} \end{bmatrix} \right\rangle \\ \text{DTR} & \begin{bmatrix} \textit{aux-verb-lex} \end{bmatrix} \end{bmatrix}$

b. $\begin{bmatrix} \textit{neg2-lex-rule} & \\ \text{SYNSEM|HEAD|FORM} & \textit{negform} \\ \text{DTR} & \begin{bmatrix} \textit{lexical-verb-lex} \end{bmatrix} \end{bmatrix}$

4.2.2 infl-head-neg

In this negation type, an inherently negative auxiliary verb is present and the lexical verb is marked with a required negative affix. I have not yet found a language with sentential negation of this type. Yet, schematically, such a construction looks as in (25):

(25) np neg1.aux iverb-neg2.

The feature structures used to model this negation type are a combination of those that we have already seen. The negative auxiliary will contribute negative semantics as in (18), but will also have to require the presence of the *-neg2* affix on its complement through the FORM feature (as in 24a), and the grammar will have to contain a rule such as (24b) to introduce the negative affix to the lexical verb and constrain its FORM value.

4.2.3 infl-comp-neg

This type has been widely discussed in the literature, as for example in French [fra] (Dryer, 2005) (26) (as analyzed by Kim and Sag 2002).

(26) Je ne-vois pas la lune
1SG NEG1-see.1SG NEG2 the moon
I do not see the moon. [fra]

To make this analysis work, the complement of the finite verb will contribute the semantics. To enforce the requirement of *neg2*, this rule will have to place an element on the finite verb's complements list. For French, the additional complement is placed at the front of the list, so we don't run into any problem finding the length of the list. The complement-changing constraint necessary to create a French-like additional verbal complement is shown in (27).

(27)
$$\begin{bmatrix} \textit{neg1-lex-rule} & \\ \text{COMPS} & \left\langle \left[\textit{neg-adv} \right] \right\rangle \oplus \boxed{1} \\ \text{DTR} & \left[\text{COMPS} \quad \boxed{1} \right] \end{bmatrix}$$

4.2.4 infl-mod-neg

In this type, sentential negation is marked by verbal inflection, and a modifier is also present.

The case of Ma [msj] (Dryer, 2005; Tucker and Bryan, 1966, 130) (28) may present an example of such a construction. In Ma, sentential negation is realized by the lexical verb being inflected by the prefix *tá-*, and an obligatory element which is inflected for agreement with the subject being placed at the end of the VP. Tucker and Bryan refer to this element as a "postposition inflected for person", but here I suggest that we may treat this element as a post VP modifier which combines with a VP marked by negation.

(28) tá-mù-sùbù-li nɔ́ŋgbɔ́ nyɔ̀
NEG-1SG-eat-PST meat NEG.1SG
I did not eat meat. [msj]

Given the four types of elements marking sentential negation in the model under development, we might ask whether *nyɔ̀* is a negative auxiliary verb or a negative modifier. The interaction of this element with the subject and the fact that the element combines with a VP might suggest a verbal analysis. But in Ma, auxiliary verbs are placed before lexical verbs, so treating this as an example of *infl-head-neg* would create an an idiosyncratic rule with respect to the auxiliary system of the language. However, treating the

obligatory second element as a modifier avoids unnecessary complication of the auxiliary verb system and has no problem accommodating subject agreement. Modifiers in HPSG traditionally have access to syntactic information about the heads they combine with via a HEAD feature, their MOD list.

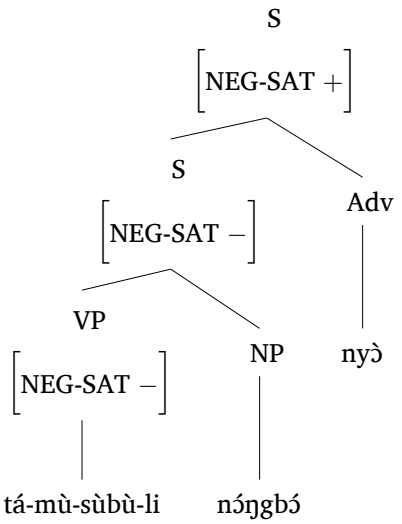
We can either add the negative semantic relation via the inflectional lexical rule which attaches to the finite verb or we can have the negative adverb provide it. Because semantically empty free morphemes are costly, placing the semantics on the negative adverb is preferred. To create the dependency between the inflectional marker of negation and the post VP modifier, an additional feature must be introduced. This *luk*-valued[8] feature is termed NEG-SAT[9] and is defined on *synsems*. The root condition is amended to require that grammatical sentences are [NEG-SAT *na-or-*$+$] and most phrase structure rules are annotated such that the value of NEG-SAT is passed up via the head-path. The lexical rule which introduces negation on the finite verb sets NEG-SAT to $-$. Finally, a subtype of head-modifier rule is defined to attach the free negator to a VP which is [NEG-SAT $-$] and create a resulting phrase which is [NEG-SAT $+$]. In this way, the lexical rule which attaches negation to the finite verb can only appear in a grammatical sentence which also picks up the secondary marker of negation once the VP is completed, as illustrated in (29). The contrasting approach would treat *neg2* as a complement and thus require a lexical rule to introduce the complement. By using a head-modifier rule, the complication of creating separate types of lexical rules to target verbs with different lengths of COMPS lists is avoided.

---

[8] *luk* is a three-valued type named after Polish Logician Jan Łukasiewicz. It allows $\{+, -, na\}$, as well as *na-or-*$+$ and *na-or–*, but $+$*-or–* is inconsistent.

[9] Mnemonic for "negation satisfied."

(29)



4.2.5 head-head-neg

This type is considered grammatically incoherent. One head cannot enforce the presence of another because from the perspective of the negation construction, this would redefine the second head as a dependent.

This type is hereby discarded.

4.2.6 head-comp-neg

In this type of double negation, an inherently negative auxiliary verb requires a complement. Schematically, such a construction looks as in (30).

(30) np neg1.aux iverb neg2.

On the surface, this type is similar to others we've seen above. The choice to model the *neg2* dependency as a complement or modifier will be dependent upon language specific argumentation. The schematic example shown in (30) can be modelled using a negative auxiliary as in (18), with the added requirement on the COMPS list for the negative particle. Note that in the cases where the we are adding a complement to a negative auxiliary verb, we do not encounter the problem of finding the length of the argument list as when adding the complement by lexical rule because it is simply specified in the lexical entry for this

negative auxiliary verb type—there is no lexical rule which much apply to modify this list once the negative auxiliary verb type has been defined in the grammar.

4.2.7 head-mod-neg

This type is similar to the *head-comp-neg* but the secondary negation marker is attached through head-modifier rather than head-complement rules. On the surface, the example looks identical to (30). To invoke this type, language specific arguments about the grammatical system under consideration would have to be made. In general, considerations of parsimony go against this sort of analysis because the NEG-SAT approach described above for *infl-mod-neg* will have to be used. Given a negative head and a (free) negative dependent, the *head-comp-neg* approach is preferred. On the other hand, if syntactic tests for argument-hood fail, the NEG-SAT approach is still a viable way to handle these sorts of constructions.

The analysis is built upon a negative auxiliary verb which is also specified as NEG-SAT $-$. As in the other constructions which use this feature, the feature amounts to a (potentially long-distance) dependency upon the attachment of a negative modifier. If, in a given language, the modifier must be attached low (i.e. cannot be long-distance), then the relevant phrasal type (S, VP) can be constrained to be NEG-SAT $+$ to achieve this.

4.2.8 comp-comp-neg

In this type, negation is marked by two obligatory complements of a verb. As with the *infl-infl-neg* type described above, we can imagine two subtypes: (a) both complements are subcategorized by the same verb; (b) one complement is selected by an auxiliary and the other by a lexical verb. The case of (a) can be modelled according to a lexical rule which attaches to a verb and modifies its COMPS list (just as in 4.1.3, but requiring two additional complements rather than just a single one). If one of the complements appears at the end of the list, this sort of analysis incurs the difficulty discussed above: subtypes of the lexical rule must be posited for each valence class.

Some examples from Afrikaans [afr] (Bell, 2004; Oosthuizen, 1998; Donaldson, 1993)

(31) presents a structure which could be analyzed as a (b)-type case. The auxiliary must place a requirement on the lexical verb that it also have undergone a complement-changing lexical rule.[10]

(31) a. Hulle was nie betrokke nie
they were NEG1 involved NEG2
They were not involved. [afr]

b. Hy het nie gedink aan die ernstige gevolge nie
he has NEG1 think of the serious consequences NEG2
He didn't think of the serious consequences. [afr]

This dependency can be achieved via the engineering of a feature which is passed up the head path when a verb is negated. A head feature [NEGATED *luk*] can be introduced by a lexical rule (in this case, the same rule which introduces the verbal complement). Then the finite auxiliary will also require that its lexical verb complement be [NEGATED +].[11]

4.2.9 comp-mod-neg

In particular examples, this negation type would look similar to *comp-comp-neg*. Syntactic tests for the treatment of the secondary negator as a modifier will have to be made. We can create an analysis of this type using a lexical rule to introduce the *neg1* complement, and the NEG-SAT analysis (as presented above) to create the requirement that *neg2* be attached through a head-modifier rule.

---

[10]It has been pointed out that treating the second negator as a selected complement incurs the length of COMPS list problem discussed above. This is true. But one must keep in mind that the length of COMPS list problem is solvable using an articulated family of lexical rules. Secondly, the length of COMPS list problem does not invalidate the potential for the Afrikaans data to fit the *comp-comp-neg* analysis presented here.

[11]Note that the NEGATED feature presented here contrasts with the NEG-SAT approach discussed above in that NEG-SAT is not a HEAD feature. This is because the inheritance which satisfies the dependency in the NEG-SAT approach comes from a modifier, the non-head daughter of a head-modifier rule, breaking the head-feature principle.

4.2.10 mod-mod-neg

To create a construction with two required modifiers, we can adapt the NEG-SAT approach described above such that the attachment of the first negator (rather than a lexical rule) sets the phrase's NEG-SAT value to $-$, then the second negator will still go through a specialized rule which will set the value back to $+$. Because only clauses which unify with [NEG-SAT *na-or-*$+$] are licensed, this approach will require *neg2* to appear whenever *neg1* does (although there may be the intervention of other modifiers and complements, as expected for head-modifier constructions).

4.3 Summary

In this chapter, I have laid out a comprehensive space of negation analyses which are hypothetically useful to a grammar engineer in creating an HPSG for a natural language. In summary, figure 4.1 reprises figure 3.1 with annotations to indicate which negation types have subtypes, which are attested (potentially) and which are merely predicted, and which are discarded

The next step is to present a system for selecting amongst these analyses—an implementation within the Grammar Matrix customization system and a questionnaire subpage for controlling it. The next chapter presents such a system.

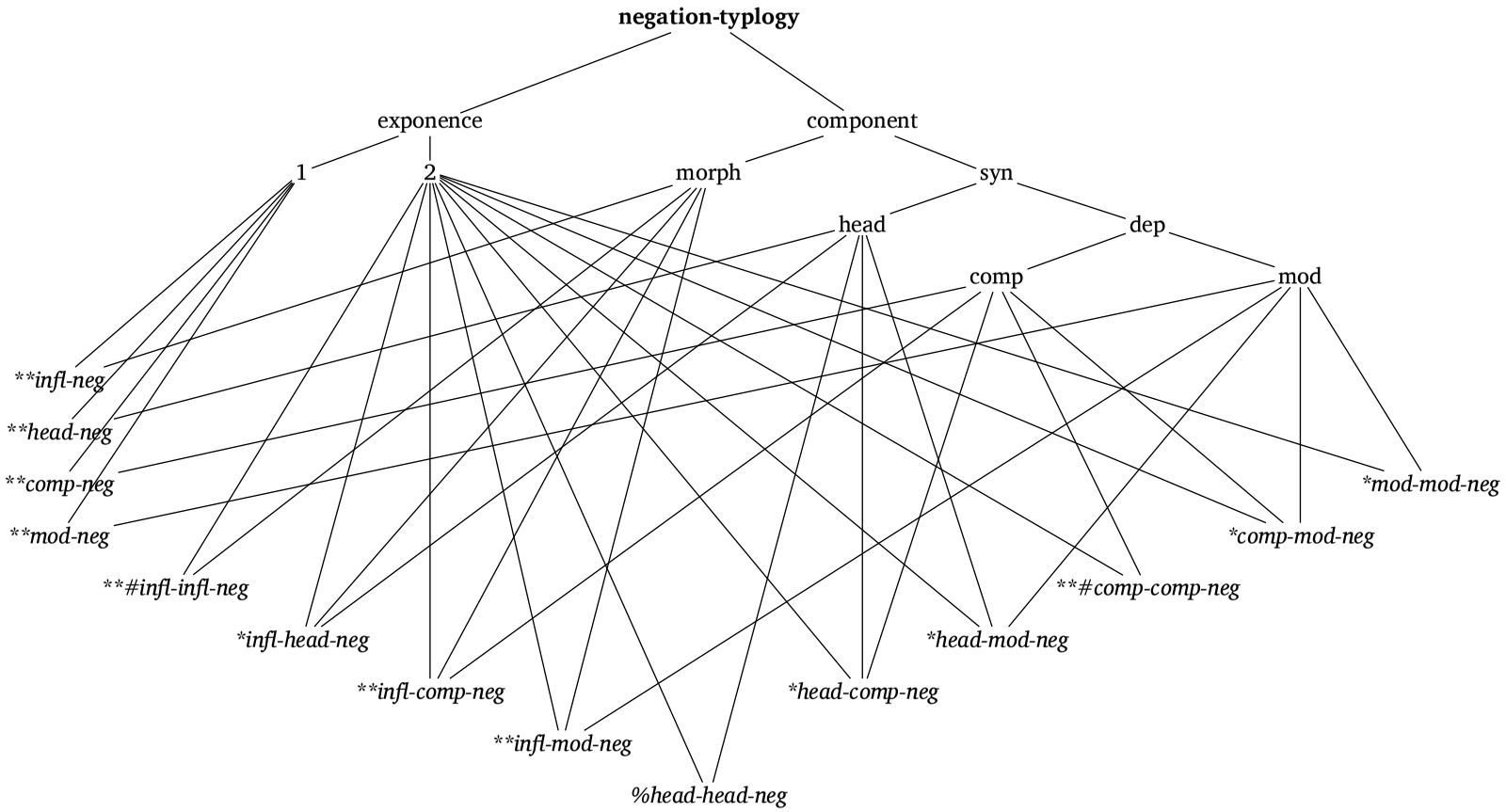


Figure 4.1: Negation model annotated: **attested, *predicted, %discarded, #has proposed subtypes

Chapter 5

IMPLEMENTATION

In this chapter, I describe the implementation and integration of the types proposed in the last chapter into the customization system. The first part of this chapter discusses the outward-facing aspects of my implementation—user interaction, how information is solicited and what sorts of prompts and feedback are offered. The latter part looks at the inner workings of the library—how analyses are broken down into TDL statements and assembled into a coherent component of a grammar based on input. I present an overview of the inventory of the grammatical types (along with their parameters of variation) which are output by the library stepping through particular properties of individual types at the level of detail considered when they were implemented.

5.1 Facing outward: user interaction

As discussed in chapter 2, the questionnaire is divided into subpages and "sentential negation" is already included amongst these. So, at first consideration, users are expected to answer questions about sentential negation on the sentential negation subpage. However, a major aspect of each of the negation constructions included in the model presented in chapters 3 and 4 is that they are comprised of a collection of grammatical objects which work together. Some of the constructions include specialized phrase structure rules, but each of them defines one or more lexical items (either morphological rules or free standing forms) which are crucial to the expression of the construction. In organizing a strategy for user input, it's important to recognize that the customization system already comes with lexicon and morphology subpages that implement a sophisticated interface for soliciting information about lexical items from users. Therefore, insofar as possible, I want to take advantage of this interface, and direct users to enter information about negative lexical items on the lexicon and morphology pages.

In order to allow this, I needed some mechanism for users to indicate which lexical items entered on the lexicon and morphology pages correspond to the negation analysis selected on the sentential negation subpage. The choice was made to use a "customization feature". This strategy directs users to select a negation analysis on the sentential negation subpage. Analyses which expect negative auxiliary verbs or negative bound inflectors unlock the feature [negation plus] for use on the lexical input pages. Then, as users define the hierarchy of lexical and morphological items (including those which mark negation) on the corresponding lexical input pages, they pick out those which are involved in a negation strategy by selecting [negation plus] and in this way indicate to the negation library that these items are in need of special processing. Figures 5.1 and 5.2 show the selection of a negation strategy on the sentential negation page opening up the possibility of marking lexical items as [negation plus].

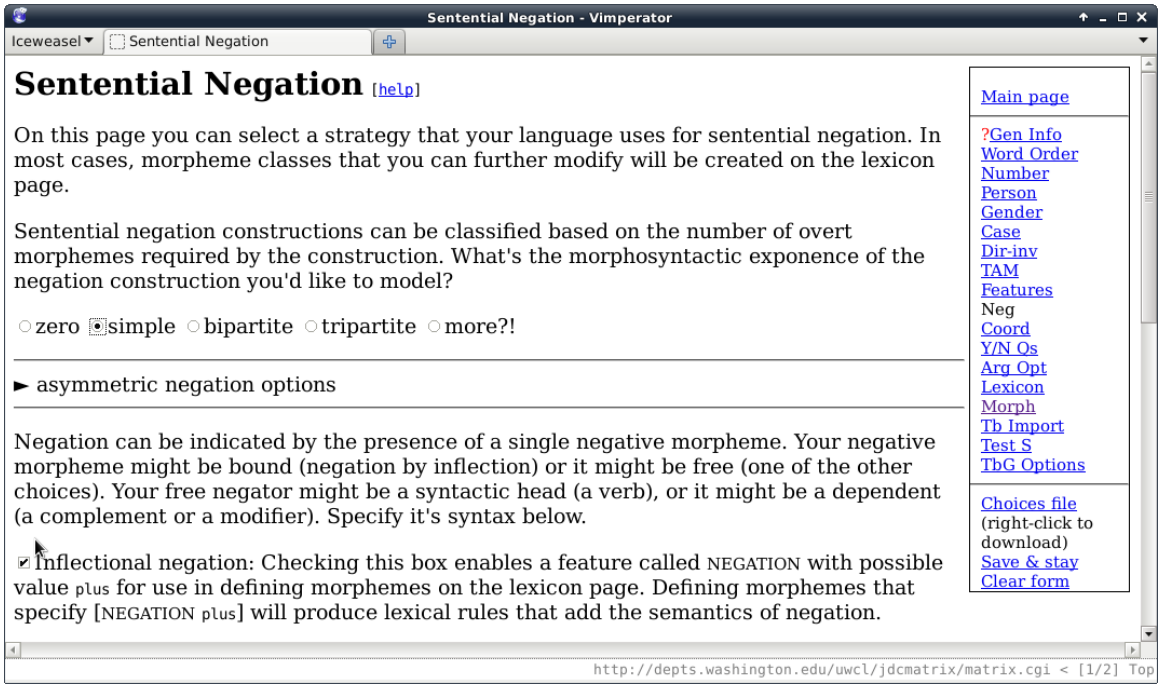


Figure 5.1: Selection of simple negation by inflection on sentential negation subpage

The "meaning" of [negation plus] on a lexical item is dependent on the negation anal-

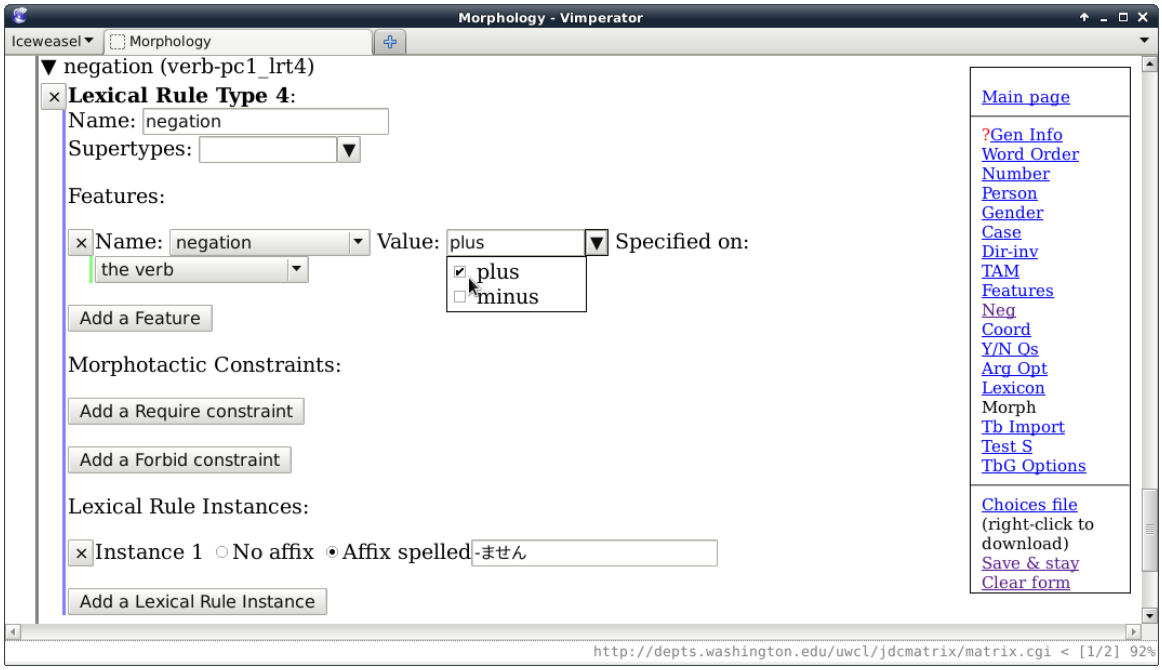


Figure 5.2: Indicating [negation plus] on a morphological rule

ysis selected on the sentential negation subpage. For example, if a user selects 'simple', 'negation by inflection' as the strategy, marking [negation plus] on a bound inflector will add negative semantics to that lexical rule type. Whereas with the strategy 'bipartite', 'infl-comp negation', marking [negation plus] on a bound inflector will create a lexical rule which requires the presence of the negative complement (by modifying the COMPS list of its verbal host) and this inflectional rule will not include negative semantics.

Because the customization system's lexical entry subpages do not yet offer support for any adjectives, adverbs or other modifiers, when these lexical items are required in a negation construction, information about the item is solicited directly on the negation subpage itself. This is shown in figure 5.3

Choosing the value 'plus' for the negation customization feature implies that 'minus' may also be significant. In fact, Miestamo 2005 suggests that a proper analysis of negation in many languages needs to take into account the fact that not all lexical items are compatible with negation marking. I have engineered the negation library so that users can mark

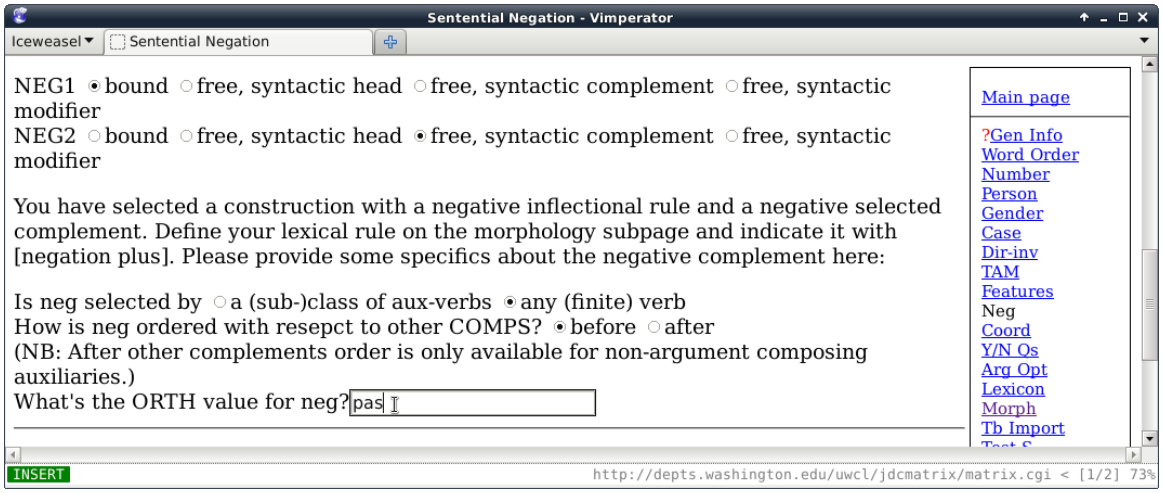


Figure 5.3: Specifying parameters for a negative selected complement on the sentential negation subpage

items [negation minus] in order to accomplish this. Lexical items marked [negation minus] in the lexicon page are made incompatible with negation using a HEAD feature.[1] The customization feature [negation minus] is made available for any negation construction by enabling an option on the sentential negation subpage (figure 5.4), and is automatically available whenever a negation strategy requires [negation plus].

To summarize the overall strategy for user interaction:

1. users select a negation strategy on the sentential negation page
   - users select exponence 1,2
   - then, depending on exponence, the appropriate subset of the 13 strategies of the model are made available for selection
2. depending on their choices, users are prompted to enter information about the lexical items involved in negation either on the lexicon page (for negative auxiliaries), on the

[1]This use of this feature was proposed for Basque [eus] negation patterns in Crowgey and Bender 2011.

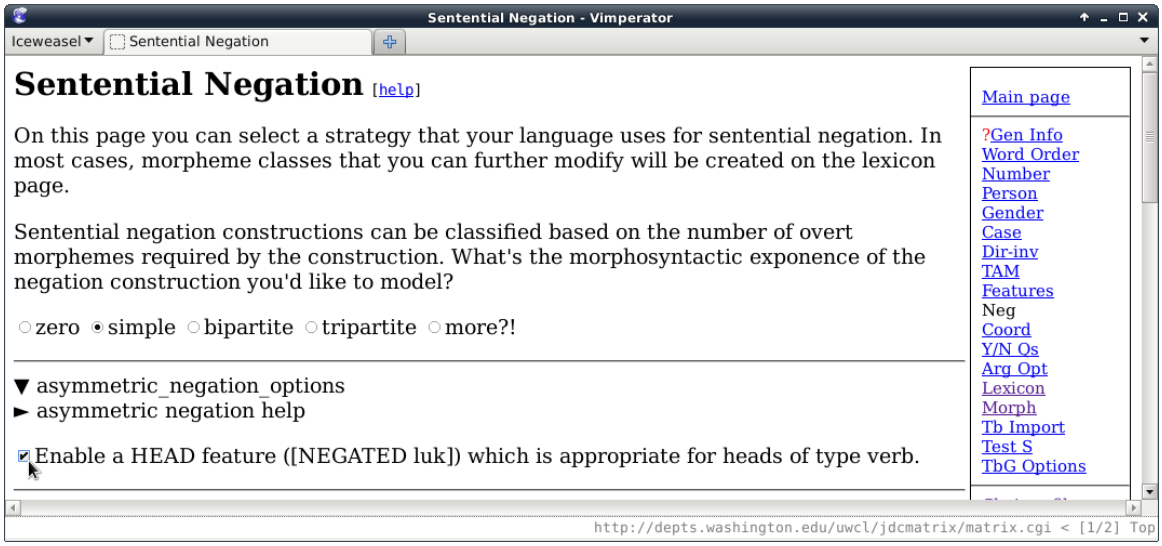


Figure 5.4: Enabling a HEAD feature for use with negation

morphotactics page (for bound material) or the negation page itself (for complements and modifiers)

Questionnaire validation (described in chapter 2) provides feedback on choices as users complete the questionnaire. For example, if the user selects a strategy involving negative auxiliaries, they must also select that their language includes auxiliaries on the word order page and they must fully define a negative auxiliary class on the lexicon page.

5.2 Behind the scenes: definitions of types and TDL assembly

Implementing a grammatical analysis as a part of the customization system means not only providing a way for users to call up that analysis while filling out the questionnaire (as described above) but also creating the definitions (constraints) of the types employed in the analysis and including them in the output grammar.

As described in Chapter 2, grammar customization is the process of taking input in the form of a choices file and deriving types based on the choices, combining this with the "core grammar" and packaging the whole as a series of files ready for loading in a

grammar development environment or other processing engine. In more detail, the process is directed by a customization script which calls each of the libraries sequentially so that the choices can be evaluated by the library and it can create its types and include them in the grammar. In practice, at least for the negation library, the components which the sentential negation page needs to include in the grammar are sometimes created by a separate subsystem (a separate library) of the customization system. For example, selecting simple negation by inflection in the sentential negation page requires interaction with the morphotactics system in order to output negative lexical rules which are integrated with the morphotactics of the language. Similarly, negative auxiliaries (for the most part) need to behave as other auxiliary verbs of a given language, and therefore need to be integrated with the word-order/auxiliaries library.

On the whole, however, implementing a negation analysis from the model described in previous chapters requires:

- deciding what grammatical features need to be added to the output grammar's feature geometry, and defining them in the grammar's type file

- deciding what lexical rules need to be added (if any) and making this information available to the morphotactics library

- deciding what auxiliary verbs need to be created, what their properties are, and making that information available to the word-order/auxiliaries library

- deciding what adverb-like (whether complement or modifier) lexical items are to be created, and adding them to the grammar files directly (along with any relevant supertypes)

- deciding what phrase-structure rules need to be added to the grammar, if any, and adding them in the rules file of the grammar directly

The negation library[2] implementation carries out these items and we can now look at the collection of types generated and tested for each analysis in the model. My expository strategy takes two tacts, first I inventory all of the grammatical objects which are used in the constructions defined in the model: lexical rules, auxiliary verbs, then complements and modifiers. After that, I discuss the library from a construction-by-construction basis, showing the relevant process for assembling that construction.

5.2.1 Overview of types

Lexical rules

In all, eight lexical rule types were needed to implement the model presented in earlier chapters. Seven of the 13 constructions analyzed in Chapter 4 require the definition of at least one lexical rule. These rules are presented here with reference to their supertypes in the Grammar Matrix core grammar, their definitions and their usage in the constructions.

basic-infl-neg is a rule for simple, inflectional negation. At a high level, this rule contributes negative semantics and attaches an affix and does nothing more. To construct this functionality, the rule inherits[3] from `cont-change-only-lex-rule`, which rule passes up all grammatical features (between mother and daughter) except CONT. CONT is constrained by virtue of the fact that generally on lexical rules (32), the CONT|RELS of the mother is the list-append of the CONT|RELS of the daughter and the C-CONT|RELS of the lexical rule. This is the basic means of semantic contribution for lexical rules,[4] so this constraint is defined generally on the type `lex-rule` Flickinger and Bender (2003), shown in (32). Beyond what it gets from inheritance, the basic inflectional lexical rule only has

[2]It should be said that at this level of detail, the term "negation library" becomes ambiguous. On one hand, it conceptually refers to any code written to carry out the points above. On the other hand, it refers to a specific file `negation.py`, which is called by the customization script directly. However, implementing the requirements above took writing code in a number of other files, or libraries, including `auxiliaries.py`, `morphotactics.py`, `features.py` and `deffile.py`. Thus the concept of a negation library is actually distributed amongst the computer files of the customization system's internal architecture.

[3]See Chapter 2.2 for discussion of types as organized into an inheritance hierarchy.

[4]This is the same approach to semantic compositionality as that which is applied to phrase structure rules.

to specify its semantic constraints (33).[5]

(32)
```
lex-rule := phrase-or-lexrule & word-or-lexrule &
      [ SYNSEM.LOCAL.CONT [ RELS [ LIST #first,
                                   LAST #last ],
                           HCONS [ LIST #hfirst,
                                   LAST #hlast ] ],
        DTR #dtr & word-or-lexrule &
            [ SYNSEM.LOCAL.CONT [ RELS [ LIST #first,
                                         LAST #middle ],
                                  HCONS [ LIST #hfirst,
                                          LAST #hmiddle ] ] ],
        C-CONT [ RELS [ LIST #middle,
                        LAST #last ],
                 HCONS [ LIST #hmiddle,
                         LAST #hlast ] ],
        ARGS < #dtr > ].
```

(33)
```
basic-infl-neg-rule = := cont-change-only-lex-rule &
       [ C-CONT [ HOOK [ XARG #xarg,
                         LTOP #ltop,
                         INDEX #ind ],
                  RELS <! event-relation &
                          [ PRED "neg_rel",
                            LBL #ltop,
                            ARG1 #harg ] !>,
                  HCONS <! qeq &
                          [ HARG #harg,
                            LARG #larg ] !> ],
         SYNSEM.LKEYS #lkeys,
         DTR [ SYNSEM [ LKEYS #lkeys,
                        LOCAL [ CONT.HOOK [ XARG #xarg,
                                            INDEX #ind,
                                            LTOP #larg ],
                                CAT.HEAD verb] ] ] ].
```

val-and-cont-change-infl-neg-rule is a rule used in the double inflectional negation construction (*infl-infl-neg*, Chapter 4.2.1). It inherits from `val-and-cont-change-lex-rule` because it not only contributes semantics, but also requires the *negform* value on its combinatorand's FORM feature. This rule includes the specifications of `basic-infl-neg-rule`

[5]This rule is essentially a TDL description of the inflectional negation AVM shown above in (14).

(without the supertype) shown in (33), but additionally has the restrictions shown in (34). In the *infl-infl-neg* construction, the inflectional rule which carries negative force attaches to an auxiliary verb and requires that its verbal complement is also inflected by a negator. The secondary lexical rule used in this construction is user defined but can be termed `form-change-neg-rule` (described more directly below).

(34)
```
val-and-cont-change-infl-neg-rule:=
    [ SYNSEM.LOCAL.CAT.VAL.COMPS.FIRST.LOCAL
        [ CAT [ VAL #val,
                HEAD.FORM negform ],
          CONT.HOOK #hook ],
      DTR.SYNSEM.LOCAL.CAT.VAL.COMPS.FIRST.LOCAL
        [ CAT.VAL #val,
          CONT.HOOK #hook ] ].

  val-and-cont-change-infl-neg-rule :+
    [ SYNSEM.LOCAL.CAT.HEAD.AUX + ].'
```

form-change-neg-rule This morphological rule is defined by users on the morphotactics page and constructed entirely using the existing system's morphology interact. This rule is defined in order to complete the *infl-infl-neg* negation analysis. It inherits from `add-only-no-ccont-lex-rule` which constrains all features on the mother to be subtypes of their correspondents in the daughter. The rule sets the FORM value of a verb to *negform* as shown in (35).

(35)
```
form-change-lex-rule := add-only-no-ccont-lex-rule &
    [ SYNSEM.LOCAL.HEAD.FORm negform ].
```

comps-change-neg-rule is used in the *comp-neg* and *infl-comp-neg* analyses of Chapter 4.1.3 and 4.2.3. It inherits from `val-change-only-lex-rule` because it does not contribute negative semantics itself, but only modifies the COMPS value such that a negative complement will be required. Some of the constraints defined on this rule are shown in (36).

(36)
```
comps-change-neg-rule:= val-change-only-lex-rule &
        [ SYNSEM.LOCAL [ CAT.VAL [ SPR #spr,
```

```
                                    SPEC #spec,
                                    SUBJ #subj,
                                    COMPS < canonical-synsem &
                                     [ LOCAL.CAT.HEAD [ NEGATED +,
                                MOD < [ LOCAL.CONT.HOOK #hook ] > ] ]
                                                . #oldcomps > ] ],
          DTR.SYNSEM.LOCAL [ CAT.VAL [ SPR #spr,
                                       SPEC #spec,
                                       SUBJ #subj,
                                       COMPS #oldcomps ],
                                CONT.HOOK #hook ] ].
```

neg-sat-trigger-rule This rule is used in the *infl-mod-neg* construction of Chapter 4.2.4 to require the presence of the negative modifier. It inherits from `add-only-no-ccont-rule` because its only job is to set [NEG-SAT $-$] (recall that phrase structure rules are also amended to pass up the value of NEG-SAT in this analysis). Shown in (37).

(37)
```
neg-sat-trigger-rule := add-only-no-ccont-rule &
      [ SYNSEM.NEG-SAT - ].
```

val-change-neg-rule-1 this rule inherits from `val-change-only-lex-rule` and is used to add the first negative complement to its verb's COMPS list in the (b) type *comp-comp-neg* analysis of Chapter 4.2.8. It accomplishes this as shown in (36), but can only attach to auxiliary verbs and also constrains the HEAD value of its verbal complement to be [NEGATED $+$] (requiring negation marking on its complement). Furthermore, this rule picks out the contentful negative modifier as a complement by ensuring that its RELS list is not empty (38).[6]

(38)
```
val-change-neg-rule-1 :=
      [ SYNSEM.LOCAL.CAT.VAL
         [ SPR #spr,
```

[6]This stipulation is not included in order to pick out the negative modifier from other modifiers generally, the two negative word classes are already distinguished from other modifiers by their use of [NEGATED $+$]. The constraint on RELS here is used because the dummy negative word is already required to constrain its RELS list to the empty list in order to allow proper semantic composition to continue after it is attached. Seen in this light, it is reasonable to use the same constraint to pick out the contentful modifier in this context. However, other features are also available to do the job (targeting MOD, for example should also work).

```
      SPEC #spec,
      SUBJ #subj,
      COMPS < canonical-synsem &
             [ LOCAL [ CAT.HEAD [ NEGATED +,
                                  MOD < [ LOCAL.CONT.HOOK #hook ] > ],
                       CONT.RELS <! arg1-ev-relation !>   ] ] .
             [ FIRST [ LOCAL [ CAT [ VAL #v,
                                     HEAD verb & [ NEGATED + ] ],
                                     CONT #c ],
                               NON-LOCAL #nl ] ] > ],
  DTR verb-lex &
      [ SYNSEM.LOCAL
          [ CAT [ VAL [ SPR #spr,
                                SPEC #spec,
                                SUBJ #subj,
                              COMPS.FIRST [ LOCAL [ CAT.VAL #v,
                                                    CONT #c ],
                                                NON-LOCAL #nl ] ],
                                HEAD.AUX + ],
                CONT.HOOK #hook ] ] ].
```

val-change-neg-lex-rule-2 This rule is the other lexical rule used in the *comp-comp-neg* construction of Chapter 4.2.8. It inherits from `cat-change-only-lex-rule` because it is used to add a dummy negative complement to its verb's COMPS list (modifying VAL) and also flips HEAD feature NEGATED's value to + (modifying HEAD). This rule's definition is presented in (39).

(39)
```
val-change-neg-lex-rule-2 :=
  [ SYNSEM.LOCAL.CAT
         [ VAL [ SPR #spr,
                 SPEC #spec,
                 SUBJ #subj,
                 COMPS < canonical-synsem &
                      [ LOCAL
                          [ CAT.HEAD
                              [ NEGATED +,
                            MOD < [ LOCAL.CONT.HOOK #hook ] > ],
                             CONT.RELS <! !> ] ] . #oldcomps > ],
           HEAD verb & [ NEGATED +,
           AUX - ] ],
  DTR verb-lex &
      [ SYNSEM.LOCAL [ CAT
```

```
            [ VAL [ SPR #spr,
                    SPEC #spec,
                    SUBJ #subj,
                    COMPS #oldcomps ],
              HEAD.AUX - ],
      CONT.HOOK #hook ] ] ].'''
```

val-change-neg-sat-trigger-rule This rule is used in the *comp-mod-neg* construction of Chapter 4.2.9. It modifies its verb's COMPS list to require a negative complement and also sets [NEG-SAT −] in order to require that a negative modifier also appear. Its definition comprises the relevant portions of the other rules shown above (modifying NEG-SAT as in (37) and the COMPS list of the verb as in (34), (38) and (39).

Auxiliary verbs

In most cases, a negative auxiliary verb is not expected to be syntactically different from the rest of a given language's auxiliary verbs. In general, auxiliary verbs are created for use in grammars by the word-order/auxiliaries library (Fokkens, 2010) and defined by choices given in the lexicon subpage. Auxiliaries which contribute a predicate are already defined to create a scopal relationship[7] with the lexical verb they combine with. So, for all but one exception, the negative auxiliaries output by the negation library all take advantage of this existing system and only have to specify a string name for the contributed predicate.

The exception regards the negation constructions in the model which fall under *head-comp-neg* and *head-mod-neg* (Chapter 4.2.6-7). These auxiliaries have to be specially created because they have particular properties which are not provided by the auxiliary library as it stood. The *head-comp-neg* analysis requires an auxiliary which also selects for an extra "dummy" argument in addition to a verbal complement. This verbs are specified as such in the lexicon. This dummy argument is the *neg2* element of the bipartite negation construction. In the case of the *head-mod-neg* construction, [NEG-SAT −] must be specified on the negative auxiliary in the lexicon. Because of the extra constraints on these auxiliaries, I modified the word-order/auxiliaries library's procedures to take special

[7] See section 2.2.2 for details of the representation.

steps when head-comp negation is selected and the feature [negation plus] appears on an auxiliary verb.

To illustrate, in (40) I present part of the definition for the negative auxiliary in the *head-comp-neg* analysis when the negative complement appears after the verbal complement. This shows the use of the HEAD feature NEGATED to pick out the negative complement from the lexicon and its placement on the COMPS and ARG-ST lists.

(40)
```
neg-aux := basic-two-arg &
          [ SYNSEM.LOCAL.CAT.VAL
                     [ SUBJ < >,
                       COMPS < #comps,
                       #negcomp &
                        [ LOCAL.CAT.HEAD.NEGATED + ] > ],
            ARG-ST < #comps &
                     [ LOCAL.CAT [ VAL [ SUBJ < >,
                                         COMPS < >,
                                         SPR < >,
                                         SPEC < > ],
                                   HEAD verb ] ],
                     #negcomp  > ].'
```

These specialized negation auxiliaries were made available for auxiliary verbs that attach to VP and S level complements, but not for lexical V complements because of the complications which arise from interactions with free word-order.[8]

Complements and modifiers

Implementing the negation library also required the creation of a number of free lexical items which vary in terms of their semantic and syntactic featural content as well as the types of (phrase-structure) rules that they combine with. When creating lexical entries for any of the negative complements or modifiers of Chapter 4, we can say that there are two types: negative modifiers and semantically empty negative dummy particles. This might

[8]Specifically, auxiliaries in the free word-order treatement of Fokkens, 2010 are arugument-composing (Hinrichs and Nakazawa, 1990), their VAL and ARG-ST lists include the nominal arguments specified on the VAL list of the lexical verb they combine with. This allows them to combine in binary rules with nominal elements subcategorized by the lexical verb—modeling free ordering of major constituents. Because, given this framework for phrase structure, the notion of adding extra arguments to these verbs is complex, this topic is set aside for future work.

be surprising because Chapter 4 contains descriptions of negative complements and negative modifiers. Yet at this level, negative complements and negative modifiers look similar. Their difference is in how they are combined syntactically—grammars that treat negative modifiers as selected complements will contain specialized head-complement rules to treat them. In the output of the negation library, both negative modifiers and negative complements inherit from an adverbial type `basic-scopal-adverb-lex` and use the MOD list to establish the proper scopal relation with the verb. This is shown in (41). When a grammar includes lexical rules to modify a verb's complement list and specialized head-complement rules to do semantic composition, the type in (41) is available for use in that grammar as a syntactic complement. Likewise, when the grammar includes head-modifier rules specific for combining a negative modifier with verbal head, this type can be used as a modifier.

(41)
```
neg-comp-lex := basic-scopal-adverb-lex &
         [ SYNSEM.LOCAL.CAT [ VAL [ SPR < >,
                                    SPEC < >,
                                    COMPS < >,
                                    SUBJ < > ],
                              HEAD [ NEGATED +,
                             MOD < [ LOCAL.CAT.HEAD verb ] > ] ] ].
```

The second type of free standing negative word output by the library are the semantically empty negative words which are required in the *head-comp-neg*, *head-mod-neg*, *comp-comp-neg*, *comp-mod-neg* and *mod-mod-neg* constructions. This type inherits from `norm-zero-arg` and has to constrain RELS and HCONS as shown in (42).

(42)
```
neg-comp-lex := norm-zero-arg &
            [ SYNSEM.LOCAL [ CAT.HEAD adv & [ NEGATED + ],
                             CONT [ RELS <! !>,
                                    HCONS <! !> ] ] ].
```

Of course in the creation of individual grammars, the negation library actual allows further parameterization of the preceding types according to linear precedence (pre- or post-head placement or either) and attachment point (lexical V, VP or S)—deriving a larger array of output possibilities for free negative words than are shown here.

Summary

	basic-infl-neg	val-and-cont-change-infl-neg	form-change-neg	comps-change-neg	neg-sat-trigger	val-change-neg-rule-1	val-change-neg-rule-2	val-change-neg-sat-trigger	basic-aux-with-pred	aux-with-dummy-arg	scopal-mod-neg	semantic-dummy
neg type	lex rules								aux		modifier	
infl-neg	x											
head-neg									x			
comp-neg				x							x	
mod-neg					x						x	
infl-infl-neg		x	x									
infl-head-neg			x						x			
infl-comp-neg				x							x	
infl-mod-neg					x						x	
head-comp-neg										x		x
head-mod-neg										x	x	
comp-comp-neg						x					x	x
comp-mod-neg							x				x	x
mod-mod-neg								x			x	x

Figure 5.5: Grammatical objects and the constructions they appear in

To summarize this section, figure 5.5 shows an overview of which grammatical objects appear in which negation analysis of the model. The columns of this chart show the grammatical objects relevant to negation which are stored in the negation library. They are assembled into the sentential negation constructions of the model of Chapter 3 and 4 as

shown by their correspondence to the rows of the chart, indicated by 'x'.

5.2.2 Overview of individual analyses

In this section, I step through the types of the library again, this time focusing on the procedure carried out. For each type, I discuss the choices which select it and the logic which is executed in assembling the construction.[9]

Simple types

infl-neg In the case of simple negation by inflection, a user has indicated that exponence = 1 and that an inflectional negation strategy is desired. Questionnaire validation (Chapter 2.1.3) ensures us then that at least one lexical rule instance is marked with the customization feature [negation = plus]. Then in the computer file `negation.py`, the script identifies items marked `negation=plus` and assigns lists any core-grammar supertypes needed for the relevant negation type, these supertypes are processed and applied to the lexical hierarchy by the morphotactics subsystem. In simple negation by inflection, the relevant supertype from the core grammar is `cont-change-only-lex-rule`. Next the value of negation = plus is modified to one that more specifically picks the relevant lexical rule type from the eight given above. This is done because yet another subsystem is responsible for compiling the TDL types and printing them out as files. This other subsystem (`features.py`) uses the modified feature value to know which type to output.[10]

neg-aux Assembling the grammatical types relevant to simple negation by auxiliary verb (Chapter 4.1.2) is easy from the standpoint of the negation library. This is because the word-order/auxiliaries library (Fokkens, 2010) provides an implementation for contentful lexical verbs which can be reused. The word-order/auxiliaries library and the lexicon

[9]For each of the following subsections, the reader is referred back to the corresponding section of Chapter 4 for details on the construction.

[10]The distributed nature of this implementation is common to all of the analyses presented here and stems the from nature of the customization system's interacting library architecture. For the most part, the following discussion abstracts away from this level of detail, focusing instead on the logic required more so than the details of the computation.

page already provide all the necessary apparatus such as determining the interaction with case or other agreement and word order. Therefore, the only work undertaken for simple negative auxiliaries was the implementation of an "autofill" functionality which creates a negative auxiliary on the lexicon page when a user selects simple negation by auxiliary on the sentential negation page. The user is then prompted by the validation system to complete the lexical entry for the negative auxiliary class.

comp-neg When a user selects simple negation by complement and fills out the details about the lexical item (pre/post other comps, spelling), the negation library has to define several grammatical objects for inclusion in the grammar.

- the [NEGATED *luk*] feature must be added to the definition of the type *head*. This feature provides the criterion for picking out the negative lexical item as the complement of the verb

- The lexical type `neg-adv-lex`[11] must be added to the grammar file. This type inherits from the core-grammar type `basic-scopal-adverb-lex`.

- The lexical rule which modifies the COMPS list of the verb to add the negative complement[12] must also be defined in the grammar file.

In order for the grammatical definitions to be useful, a lexical instance along with proper spelling and predicate value must be created in the lexicon file for the negative complement. Likewise, for the lexical rule, an instance has to be defined in the in inflectional rules fule.

mod-neg When a user selects simple negation by modifier and fills out the details about the lexical item (pre/post heat attachment (or either), spelling, attachment target (V, VP, S)), the negation library has to carry out the following procedure:

---

[11] Shown in (42), above.

[12] Shown in (37).

- The type definition for the negative adverb [13] must be added to the grammar file and an instance provided in the lexicon file.
  - The value of POSTHEAD on the adverb type defintion has to be parameterized according to user input.[14]
  - The type of constituent modified by the negative adverb is also defined on the adverb type according to the user's choices.[15]
- In order to combine the adverb with other constituents, head-modifier rules must be added to the grammar's rules file.

Bipartite types

infl-infl-neg When a user selects bipartite negation with two inflectional markers, the questionnaire expects that at least one lexical rule is marked [negation plus] in the morphotactics input page and that there should be a second lexical rule which marks [FORM *negform*] on its head. This is enforced by questionnaire validation. The rule marked [negation plus] is processed by the customization script to add negative semantics and to set up the dependency between the rules.

Because validation ensures that the user has created the rule which inflects a verb to *negform*, the second rule is created by the morphotactic system. Users will specify the rule's position class and spelling and grammatical features and the morphotactics system organizes this information into a set of TDL statements.

infl-head-neg When a user selects bipartite, infl-head negation, all the relevant specifics of the negative auxiliary can be set up using the questionnaire—the negation library itself does not need to specify any types to be added to the grammar. Instead, the negation

[13]Shown in (42).

[14]Whether a modifier attaches pre/post head or either is constrained by the value of an HPSG feature CAT|POSTHEAD which is also accessed by *head-modifier* and *modifier-head* phrase-structure rules. When POSTHEAD is unconstrained, the modifier can attach to the left or right of its head.

[15]This is accomplished by constraining MOD|FIRST|LOCAL|CAT|VAL for S and VP attachment and by constraining MOD|FIRST|LIGHT for modifiers that attach to lexical verbs.

library's implementation provides guidance about creating this construction using the lexicon page. An "autofill" functionality was built which creates a negative auxiliary verb type with the FORM *negform* requirement are already filled in on the lexicon page.

The user must also create an inflectional rule to set a lexical verb's FORM to *negform*. Validation ensures that this is the case.

To create a bidirectional implication between the auxiliary verb and the secondary inflector, the lexical rule which is created to inflect for *negform* must be in the same position class as a rule which inflects for an incompatible FORM value and the position class must be required. This requirement is not enforced in validation because the user may legitimately wish to allow one of the two negation markers to occur independently.[16]

When this is carried out, a verb can only be fully inflected if it is either in *negform* or in the incompatible FORM which will prevent it from being selected by the negative auxiliary.

infl-comp-neg When a user selects bipartite negation with an inflectional marker and one verbal complement, the negation library carries out the following procedure:

- As in simple *comp-neg*, the [NEGATED *luk*] feature to the definition of *head* because this feature provides the criterion for picking out the negative lexical item as the complement of the verb.

- The type definition for the negative complement[17] has to be added to the grammar file.

- The spelling and predicate must be specified on a lexical instance in the lexicon file

- The type definition for the lexical rule which modifies the COMPS list[18] must be added to the grammar file and provided an appropriate instance (which will be inflecting, in this case) in the inflectional rules file.

---

[16] cf. discussion of Mupun [sur] in Dryer, 2005

[17] Shown in (42).

[18] Shown in (37).

Implementation reveals that this negation construction is identical to simple *comp-neg* in that both constructions require the inclusion of the same number and type of grammatical items, with the same dependencies between them. The difference is that in the bipartite construction (*infl-comp-neg*) the lexical rule which modifies the COMPS list of the verb is inflecting, while in the simple construction (*comp-neg*) it is phonologically null.

infl-mod-neg The *infl-mod-neg* analysis is similar to *infl-comp-neg* and *comp-neg* in that it requires the definition of a lexical rule to modify a verb's grammatical features and a negative adverb type must be included. However, it differs in that the lexical rule does not change the COMPS list, but instead sets [NEG-SAT $-$]. To assemble the construction, the following procedure is carried out:

- The [NEGATED *luk*] feature has to be added to the definition of *head*, because in this construction, this feature is used to pick out the negative modifier as the non-head-daughter of the specialized phrase structure rule which resets NEG-SAT.

- The feature [NEG-SAT *luk*] must be defined as appropriate for the type *synsem*

- The definition of `basic-verb-lex` must be modified so that verbs start out [SYNSEM|NEG-SAT `na-or-+`].

- clause types must also be constrained [SYNSEM|NEG-SAT `na-or-+`][19]

- Phrase structure rules are decorated to pass up the value of NEG-SAT. For example, head-complement rules are ammended in the grammar file as shown in (43).

- The lexical rule which creates the NEG-SAT dependency[20] must be added to the grammar file.

[19]The type *clause* may not be appropriate for all languages, depending on the interaction of negation with embedding. Grammar developers can specify this feature on a different grammatical type if this is the case for a particular langauge.

[20]Shown in (38).

- The specialized head-modifier rules which attach the negative adverb to a V,VP or S type constituent and reset [NEG-SAT +] must be added to the grammar file and provided with instances.

- The negative adverb type definition[21] has to be included in the grammar file and an instance provided.

(43)
```
basic-head-comp-phrase :+
            [ SYNSEM.NEG-SAT #ns,
              HEAD-DTR.SYNSEM.NEG-SAT #ns ].
```

head-comp-neg For this analysis, implementation requires modifying the procedures of the word-order/auxiliaries library in order to create negative auxiliaries which also subcategorize for a dummy negation element in addition to their other complements. The rest of the procedure for setting up this construction is as in other negation constructions implemented here which include a free negative dependent. The semantically empty negative word and the specialized phrase structure rule which attaches it and satisfies [NEG-SAT +] must be added to the grammar (and instantiated). Finally, the definition of the HEAD feature [NEGATED *luk*] is included in the grammar file so that the negative complement can be picked out by the auxiliary.

head-mod-neg This construction also required the modification of the word-order/auxiliaries library. This time the modification does not concern valence patterns but the stipulation of the feature [NEG-SAT −] on negative auxiliaries. Other procedures are as we have seen in the other constructions which employ the NEG-SAT approach: [NEGATED *luk*] is defined on the type *head*. Phrase structure rules are decorated as in (43) to copy up the value of NEG-SAT along the head path. The definition of verbs is modified such that they start out [NEG-SAT *na-or-*+] in the lexicon. Clauses require [NEG-SAT *na-or-*+]. Finally, the type definition and lexical instance for the negative adverb are included along with customized the head-modifier phrase structure rules to attach the negative adverb to the proper constituent type and reset NEG-SAT to [NEG-SAT +].

[21] Shown in (42).

comp-comp-neg For this analysis, the negation library has to create two COMPS changing lexical rules. One of them applies to auxiliary verbs and modifies its COMPS list to require the contentful negator.[22] The other rule applies to lexical verbs and adds a semantically complement.[23] These lexical rules also access the value of the head feature [NEGATED] in order to create a dependency between the rules. In addition to the definitions of the lexical types for the two negative complements and their instances, the negation library also has to set the definition of auxiliary verbs such that they select for complements which are [NEGATED −].

comp-mod-neg This construction requires the grammatical objects associated with the NEG-SAT approach along with a COMPS changing lexical rule. Like *comp-comp-neg*, it requires two negative dependents, one which carries semantics and the other a dummy. The lexical rule included with this construction is discussed above as *val-change-neg-sat-trigger-rule* because it combines the properties of the COMPS changing lexical rule with the lexical rule that triggers NEG-SAT on a verb. Apart from the unique lexical rule which modifies a verb's COMPS list, the assembly of this construction parallels the other constructions which use NEG-SAT, requiring definitions and instances for the specialized phrase structure rules along with the lexical items.

mod-mod-neg To implement this negation strategy, the negation library has to include two negative modifiers, one which creates a NEG-SAT dependency the other which resolves it. This entails the definition of the feature and stipulations on phrase structure rules for feature passing as described above for other NEG-SAT constructions. Also, like for other constructions which include syntactic dependents, lexical types and instances must be defined and included.

---

[22] Shown in (39).

[23] Shown in (40).

5.3 Summary

In this chapter, I have presented an overview of my implementation of the negation model of previous chapters as a library of the Grammar Matrix customization system. Facing outward, the system expects users to specify details of how their language uses negation across a distributed set of subpages of the questionnaire. An overall strategy is to be selected on the sentential negation subpage but individual lexical items are to be defined on lexical input pages (when available). A "customization feature" [negation plus] is made available for users to indicate which lexical items pertain to sentential negation.

In the second part of the chapter I turned to the internal workings of the system, I have presented an overview of the grammatical objects defined and how they relate to the constructions of the model and I have stepped through the procedures defined in the customization system's internal files for outputting these analyses.

In the next chapter, I present an evaluation of this work. First, an internal validation of the analysis is described (a system of regression tests). Secondly, the library is put to use modeling a set of negation constructions for languages which were not considered directly during the construction phase of the library.

Chapter 6

EVALUATION

In this chapter, I present an evaluation of the negation library along two dimensions. The first is a validation layer on the implementation itself, a system of regression tests, described below. In the second evaluation, I put the library to use in the assembly of natural language grammars. I discuss five test cases from five languages, each from a distinct language family.

6.1 Regression tests

As mentioned in Chapter 2.1.2, regression tests are an essential component of a customization system library because they both demonstrate the functionality of the library and they ensure that that functionality will be preserved in future updates to the system. This evaluation framework was also applied in Drellishak (2009) and Saleem (2010). For the negation library, I created or modified a total of 25 tests related to the functionality of the library and its interaction with morphology and word order systems.[1]

Because there are too many tests to go through each aspect of each one individually here, I discuss only one of these tests in full detail. A regression test is made up of three main components: a choices file, a test-suite and a set of gold-standard semantic representation. A test-suite is a set of strings with associated grammaticality judgments. For example, the test-suite which is part of the test `neg-head-feature` is presented in (45). This test is intended to ensure that the negative head feature option provided by the negation library can be used to make certain affixes incompatible with negation.

(44)
```
n1 iv                       n1 tv-othersuff n2
n1 tv n2                    n1 tv-nonegsuff n2
n1 neg-iv                   *n1 neg-iv-nonegsuff
```

[1]For reference, this document refers to revision number 23263 of the Grammar Matrix software repository.

```
n1 neg-tv n2                n1 neg-iv-othersuff
n1 otherpref-iv             n1 otherpref-iv-othersuff
n2 otherpref-tv n2          *n1 neg-tv-nonegsuff n2
n1 iv-othersuff             n1 neg-tv-othersuff n2
n1 iv-nonegsuff             n1 otherpref-tv-othersuff n2
```

In the choices file which is also a part of this test (relevant portions shown in 45), the prefix `neg-` is marked with the customization feature `negation=plus` and the suffix `-nonegsuff` is marked `negation=minus`. The intended functionality is that the customization system should operate such that the grammar output by this choices file will reject the starred examples in the test suite above (44) and assign a semantic representation to the non-starred examples which matches that stored as a gold standard.

(45)
```
section=sentential-negation
neg-exp=1
neg-head-feature=on
infl-neg=on

section=lexicon
  verb-pc1_name=neg-pc
  verb-pc1_order=prefix
  verb-pc1_inputs=verb, verb-pc2
     verb-pc1_lrt1_feat1_name=negation
     verb-pc1_lrt1_feat1_value=plus
     verb-pc1_lrt1_feat1_head=verb
     verb-pc1_lrt1_lri1_inflecting=yes
     verb-pc1_lrt1_lri1_orth=neg-
     verb-pc1_lrt2_lri1_inflecting=yes
     verb-pc1_lrt2_lri1_orth=otherpref-
  verb-pc2_name=no-neg
  verb-pc2_order=suffix
  verb-pc2_inputs=verb
     verb-pc2_lrt1_feat1_name=negation
     verb-pc2_lrt1_feat1_value=minus
     verb-pc2_lrt1_feat1_head=verb
     verb-pc2_lrt1_lri1_inflecting=yes
     verb-pc2_lrt1_lri1_orth=-nonegsuff
     verb-pc2_lrt2_lri1_inflecting=yes
     verb-pc2_lrt2_lri1_orth=-othersuff
```

The other tests I have created to accompany the negation library test other functionality

associated with the library in an analogous way, varying word orders and placement of morphological markers. In the case of bipartite negation constructions, the tests ensure that a mutual-requires relationship is established between the exponents, parameterized along the lines of the model presented in Chapter 3. Each negation type in the model has one or more accompanying tests.

The existing library on sentential negation had already implemented some tests for inflectional negation and adverb attachment negation. The majority of the existing tests were inherited by the new library and continue to pass.

The entire suite of negation related tests is available as a part of the Grammar Matrix open-source software package.[2] Next, I turn to a practical evaluation of the library's usage by applying it to the job it was built for: aiding in the creation of linguistic grammars for natural languages.

6.2 Held out languages

In this section, I present the results of an experiment using the negation library to model the negation patterns of five languages which were not examined in the creation of the library. I generated this sample of languages using a pseudo random number generator and a database of linguistic descriptions (PDF documents) which was made available on the UW Linguistics computer network by the researchers on the PHOIBLE project (Moran and Wright, 2009). Because the database was created as part of a typological investigation of phonological properties, I sampled 15 PDF files and discarded those dedicated solely to phonological description or those without a description on negation patterns. After this step, I had descriptions of five languages which were adequately varied along areal and genealogical criteria. The languages I investigated are:

- Baga Sitemu [bsp]: Guinea; Atlantic-Congo
- Kashinawa [cbs]: Peru/Brazil; Panoan

[2]The Grammar Matrix customization system is available as an `svn` repository at `svn://lemur.ling.washington.edu/shared/matrix` using the account `guest`. The tests relevant to negation can be invoked by passing the option `r neg*` to `matrix.py`.

- Chinook Jargon [chn]: Coastal Pacific Northwest of North America; Creole (Waskashan, Indo-European)

- Cupeño [cup]: formerly in California (extinct); Uto-Aztecan

- Palauan [pau]: Palau; Austronesian

Each of these languages forms a small case study in the usage and applicability of the negation library to modeling natural language data. In the subsections below, I describe relevant information about each language—what aspects of the system were tested and how the system faired. In general, the procedure for each of the languages was to review the available literature and create a test-suite of positive and negative examples. Then, for each language, I developed a choices file intended to capture the phenomena relevant to the examples in the test-suite. The last step is to build a grammar from the choices file and test it out by applying it to the items of the test-suite. Full test suites and choices files for the languages described below are presented in Appendix A.

6.2.1 Baga Sitemu [bsp]

Baga Sitemu[3] is an Atlantic-Congo language of Guinea. It presents SVO word order and a noun class system reminiscent of Bantu. With respect to negation patterns, Ganong (1998) says that there are three (phonologically) invariant morphs which mark negation in particular contexts. For sentential negation one marker (a suffix which precedes object concord marking) is associated with matrix indicative clauses and a second marker (a prefix which follows subject concord marking) is used in other moods and with embedded clauses. The third marker is a free morpheme which is associated with nominal negation. The complementary distribution of the two sentential negation markers in different grammatical mood contexts provides an opportunity to test the negation library and its interaction with the tense-aspect-mood library.

---

[3]For data about Baga Sitemu I refer to Ganong 1998.

(46) a. ɛ-nɨŋk-fɛ-kɔ
3SG.SUBJ-see-NEG.IND-3S.OBJ
I did not see him. [bsp]

b. mɨ-fɔ-nɨŋk-ki
2SG.SUBJ-NEG.NONIND-see-CL5.OBJ
[that] you do not see it. [bsp]

c. *mɨ-fɔ-nɨŋk-fɛ-ki
2SG.SUBJ-NEG.NONIND-see-NEG.IND-CL5.OBJ
You do not see it.

Ganong's work also provides enough information about position classes of the morphology to model the interaction of negation with some of the other inflectional morphology she describes (specifically in my test suite, the subject and object concord elements). Thus, although Ganong's work is typical of descriptive linguistics in not providing negative examples explicitly, she does provide enough information about patterns which do not occur for me to generate negative (starred) examples to include in the test suite. Specifically, I created starred examples which illustrated the unattested co-occurrence of the two bound negative morphs (46) (from Ganong (1998, 72-73)) as well as examples with the inflectional morphology attached in an unattested order (47).

(47) a. ɛ-n-tam-ɨnɛ-fɛ
3SG.SUBJ-PHON-ABLE-RECPR-NEG
He is not well. [bsp]

b. *ɛ-n-tam-fɛ-ɨnɛ

c. *-fɛɛ-n-tam-ɨnɛ

d. *ɛ-n–fɛtam-ɨnɛ

e. *ɛ–fɛn-tam-ɨnɛ

The negation library was able to create, with an appropriate choices file, a grammar which was able to parse each of the attested examples and provided the correct semantics. It also ruled out each of the unattested sentences. The Baga Sitemu test case shows the negation library is adequate for creating multiple negation constructions in a language (and can be used to force those constructions to be mutually exclusive).

6.2.2 Kashinawa [cbs]

Kashinawa is a living Panoan language of Peru and Brazil. The PDF database I accessed for descriptions included a didactic grammar for Kashinawa aimed at readers of Brazilian Portuguese (Montag, 2004). While this is the principle work which I used to create a test-suite for sentential negation in this language, I also referred to Montag (1973, 2005) and Camargo (2002). In contrast to the didactic grammar, these other works use the Peruvian spelling system and provide morpheme segmentation and interlinear text. After studying these resources on Kashinawa, I found that the marker of sentential negation varies its form in agreement with the NUMBER value of the sentential subject (48) (Montag, 2004, 48).

(48) a. Ẽ pi-ama-ki
1SG.A eat-NEG.SG_SUBJ-DECL
I don't eat (it). [cbs]

b. ∅ Pi-abuma-ki
3A eat-NEG.PL_SUBJ-DECL
They don't eat (it). [cbs]

I therefore created a test-suite with positive and negative examples. The negative examples were made by attaching affixes in an incorrect order and by using singular subject pronouns with the plural negation marker (49).

(49) a. *pi-ki-abuma
eat-DECL-NEG.PL_SUBJ

b. Ẽ pi-abuma-ki

1SG.A eat-NEG.PL_SUBJ-DECL

The negation library was able to create, given a hand developed choices file for this data, a grammar which parses the positive examples and rules out the negative ones. The test case of Kashinawa shows that the negation library can handle inflectional markers of negation which specify other grammatical information such as PERSON, NUMBER or GENDER values of the subject.[4]

6.2.3 Chinook Jargon [chn]

Holton 2004 provides a description of this mostly extinct creole language which was spoken in the Pacific Northwest of North America. Holton's work describes two sentential negation markers in Chinook Jargon. The first is expressed as a free negator which precedes the sentence (including the required subject pronoun).[5] The second is a negative modal auxiliary of possibility which he glosses *can't* (50).

(50) a. Nayka nanich

1SG watch

I watch. [chn]

b. Wik nayka nanich

NEG 1SG watch

I don't watch. [chn]

---

[4]It is important to point out that the ability of the negation library to handle the interaction of negation markers and subject features is not due to any indication in the negation literature that this would be necessary. Instead, we have to credit the generalized construction of the lexicon and morphotactics subpages (a system designed with the fact in mind that generally languages allow morphemes to simultaneously mark multiple grammatical properties).

[5]Depending on the treatment, either subjects must be pronominal and are required or they could be inflectional markers of subject agreement on the verb. The one of the syntactic facts which bears upon the question is that in Chinook Jargon, when the subject is 3rd person, an NP can precede the (resumptive) required subject pronoun and the proper analysis for this preposed NP construction would still have to be proposed. On the other hand, the morphological analysis does not suffer from any difficulty capturing the NP, it is simply the subject when the subject is overt. The best analysis for Chinook is pursued no further here. The choices file I developed for the language treats subject marking as pronominal and does not include test cases with the preposed NP.

c. Hawkwêtl nayka nanich
NEG.CAN 1SG watch
I can't see. [chn]

I was able to create a test suite with negative examples by varying the attachment point of the free negator (51).

(51) a. *nayka wik nanich

b. *nayka nanich wik

In using the negation library to create a grammar which captures these positive and negative examples, I was able to handle the free negator by treating it as an adverb which attaches to S. However, the example with the integrated negation and modal possibility revealed a weakness of the negation library. While my system can handle negative auxiliaries, we (as of yet) have no way to specify lexical items which contribute multiple predicates in the customization system lexicon interface.[6]

The case of Chinook Jargon shows that while the negation library adequately handles adverbial negation and (simple) negation by auxiliary, negation integrated into lexical items which contribute other predicates has to be defined in a grammar by hand after customization.

6.2.4 Cupeño [cup]

Cupeño is an extinct Uto-Aztecan language of Southern California, documented extensively in Hill 2005. Cupeño exhibits a number of phenomena which are not directly supported in the customization system at this time. These range from the relatively trivial (no orthographic support for phonological attachment of forms which are syntactically positioned, i.e. clitics) to more difficult (discontinuous NPs, reduplicative morphs) and include items

[6]It should be emphasized that this is a weakness of the customization system only, not the underlying framework. Lexical items with multiple predicates (and scope restrictions between them) can be defined in TDL.

on the Grammar Matrix's list of planned extensions (finite V2 ordering with lexical verb strictly last (SOV)). What these intriguing phenomena have in common is that Hill's description of Cupeño suggests that they are all basically orthogonal to sentential negation phenomena. Sentential negation in Cupeño is expressed via the inclusion of a sentential adverb which occurs sentence initially (52) (Hill, 2005, 389).[7]

(52) a. Qay = el = pe muyax-yax
NEG = 2PL_SUBJ = IRR go.out-YAX.F
Don't you all be going outside. [cup]

b. Qay e-'ye-t-im pe'-miyax-wen
NEG DUP-theif-NPN-PL 3PL-BE-PIPL
They were not theives. [cup]

In order to create a test suite which I could hope to parse, I had to make certain abstracting modifications to the representation of these sentences. Because there is no direct support for clitic attachment, I treated the clitic complex as an auxiliary verb in 2nd position. Likewise, with the reduplicative morpheme attached to *'ye*, I was forced to change it's phonological content to "dup". I show my abstracted sentences of Cupeño in (53).

(53) a. `qay =el=pe muyaq-yax`

b. `qay DUP-'ye-t-im pe'-miyax-wen`

I also added starred examples to the test-suite by placing the negative modifier in incorrect syntactic positions, as in (54).

(54) a. `*=el=pe qay muyaq-yax`

b. `*=el=pe muyaq-yax qay`

[7] Hill explicitly states (p. 389) that examples in which the negative marker *qay* is preceded by other material clearly show signs of clefting or topicalization.

c. *DUP-'ye-t-im qay pe'-miyax-wen

d. *DUP-'ye-t-im pe'-miyax-wen qay

Once these modifications were made to the test-suite, I was able to create a choices file and grammar that models the negation processes of Cupeño. I treat *qay* as an adverb-like negator which attaches to S and precedes its head.

The test case of Cupeño shows that negation by an independent adverb works well in sentences with both auxiliaries and without (52a, b).

6.2.5 Palauan [pau]

Palauan is an Austronesian language spoken on islands of Palau. For information on Palauan, I refer to Josephs 1975. This work provides a chapter on negative sentences with many examples, as sample of which are shown (55) (Josephs, 1975, 362).

(55) a. A ngęlękek a smechęr
f child.POSS1SG AGR$_f$ sick.V
My child is sick. [pau]

b. A ngęlękek a diak lsechęr
f child.POSS1SG AGR$_f$ NEG sick.HYP.3SG_SUBJ
My child is not sick.[pau]

Morphologically, these items are complex. Josephs does not offer an interlinear glossed example, so I have filled in the second line in the examples above based on the tables and prose descriptions that are provided.[8]

In Palauan, verbs can be marked with a prefix that indicates "hypothetical" mood, this prefix also agrees with the subject in PERSON and NUMBER. Palauan negation is accom-

[8]I have glossed the "Palauan word" *a* as f, because its semantic function is unclear to me and I believe that its content is not relevant to sentential negation in this language. Josephs says that this word appears before all non-pronoun NPs. Finite Palauan verbs are marked with this word as well, so it can be seen as a kind of inflectional exponent of subject agreement. Therefore I gloss the verbal *a* as AGR$_f$.

plished using a negative auxiliary verb and this verb requires hypothetical marking on its complement.

Because Palauan morphology exhibits infixing and is sensitive to phonological constraints which cannot currently be modelled in the customization system, I created an abstraction of Palauan of the kind which would be carried out by a morphophonological preprocessing step, linearizing complex affixation to simple edge attachment, as illustrated in (56).

(56) a. `det child ind-sick`

b. `det child negaux hyp-sick`

I also added negative examples to the test suite as shown in (57). These show that it is ungrammatical for a negative auxiliary to combine with a verb that lacks mood marking, or that has indicative marking.

(57) a. `*det child negaux ind-sick`

b. `*det child negaux sick`

I then developed a choices file for this abstracted Palauan which yielded a grammar that was able to successfully analyze the test suite—parsing the grammatical examples to the correct semantic representations and failing to parse the ungrammatical ones.

The case study of Palauan shows that the negation library's treatment of negative auxiliary verbs is successful in both contributing negative semantics and to placing ancillary morphosyntactic requirements on its complement as required by the grammaticalization phenomena of a given language.

6.3 Summary

The negation library was evaluated along two dimensions: a set of regression tests provided a basic validation layer to the analyses described in this thesis and in the second

component, an experiment was performed in which the library was put to use in creating grammars for natural languages.

The suite of regression tests included with the library ensure that the library behaves as intended, both presently and as the software is extended in the future. The tests are made of up a choices file, a test suite and a gold standard semantic representation and can be run programmatically as part of the software.

The experiment with held out languages uncovered negation by inflection ([bsp],[cbs]), by independent adverb ([chn, cup]) and by negative auxiliary ([pau]). The languages also tested the interaction of negation with other grammatical phenomena such as subject agreement, mood, word order and complementation. Overall the library provided an adequate model for the majority of the data considered. The one weakness found was the lack of ability to account for a negative auxiliary verb in Chinook Jargon which also carries a modal predicate. An extension to the customization system could in principal include a treatment of auxiliaries with multiple predicates and a scopal relation, pending future research.

Chapter 7

CONCLUSION AND OUTLOOK

This thesis has presented a new library for the Grammar Matrix customization system. To accomplish this, I looked to both syntactic theory and typological surveys and proposed a finely articulated model of sentential negation which includes predicted types alongside attested ones. The syntactic analysis was implemented and tested via a system of regression tests and an external evaluation was performed by putting the library to the test on languages not considered during the creation of the library. The evaluation suggests that the library is well constructed to model natural language data on sentential negation.

While much has been accomplished, here I can list some of the items which were not addressed by this library and can be considered future research. These can be divided into two types: future work on the negation library proper and topics which were uncovered by the negation library, but which actually concern other components of the system.

In the first the first item not addressed here is support for EDGE inflection[1]. The customization system is does not yet support inflection which appears at the edge of phrases, so this type of morphosyntactic attachment could not be modeled in the negation library.

Another weakness of the library as it stands is the lack of explicit support for multiple negation constructions. Because of the interaction with the lexicon and morphology subpages and because of the usage of a single customization feature to indicate any negation strategy, modeling multiple negation constructions is not directly supported. To me, it looks like supporting multiple negation strategies would require a rethinking of the current interface between negation and the lexical pages that uses only a single interpretation of [negation plus] per customization.

A final weakness of the negation library is the underdeveloped ancillary options for "zero" and "tripartite" exponence of negation. The model proposed here allows the syn-

[1] Cf. Crysmann 2010 for discussion and references

tactic exponence of sentential negation vary between 1 and 2, but this fact suggests that a scale. So the question can should be asked whether negation constructions are attested with exponence of 0 or 3. Perhaps surprisingly, there is an attested construction of zero exponence reported in Master (1946) for Old Kannada. This construction is discussed in Miestamo (2010) from the point of view of typology, Crowgey (2012b) examines in the light of HPSG and the Grammar Matrix. Budd (2010) reports a tripartite construction in a language of Vanuatu. Incorporating such typological rarities may required a trade off with simplicity of interface, but the report that these sentential negation types exist provides a confirmation that the model presented here is on the right track.

Secondly, I turn to some aspects of the customization system which could seemingly be improved at least as indicated by their interactions with negation. The first of these is the integration of lexical items with multiple predicates into the lexical entry pages of the system. Chinook Jargon [chn] presents an auxiliary which grammaticalizes both negation and a modal of possibility (with negation having higher scope). This lexical item cannot be fully specified in the customization system as it stands. Presumably, this functionality would be generally useful and seems to be a doable task. Another possibility for improvement would be generalization of the system which creates auxiliaries with dummy complements in the *head-comp-neg* construction. As things stand, these types are only available to the negation library. Yet, it may be the case that such types would be useful to other libraries. Because of the possibility that other systems of the customization system could take advantage of such lexical items , having creating them only for negation is not ideal in terms of the system's overall elegance and parsimony.

I close by reprising the main achievements presented in this work. The Grammar Matrix customization system's library on sentential negation was improved to achieve broader coverage over the negation patterns in the world's languages. The library's types now have a foundation in typological literature. A new model of syntactic analyses for sentential negation was proposed and implemented. Existing literature treated double negation as undifferentiated, but this thesis explored logically possible subtypes for bipartite negation and proposed novel analyses. A survey of negation data found many of the predicted types to be attested, or potentially attested pending language-specific argumentation. The new

library was validated by means of a battery of regression tests which ensure the work completed here will not be undone by future modifications to the system and the library was put to the test in an experiment which drew on linguistic data from five separate language families. The new system has already been deployed on the Grammar Matrix project's web-site[2] and is available for use.

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

BIBLIOGRAPHY

Bauer, Winifred. 1993. *Maori*. Routledge.

Bell, Arthur. 2004. *How N-words move: Bipartite Negation and 'Split-NegP'*, pages 77–114. Mourton de Gruyter.

Bender, Emily M., Drellishak, Scott, Fokkens, Antske, Goodman, Michael Wayne, Mills, Daniel P., Poulson, Laurie and Saleem, Safiyyah. 2010a. Grammar Prototyping and Testing with the LinGO Grammar Matrix Customization System. In *Proceedings of the ACL 2010 System Demonstrations*, pages 1–6, Association for Computational Linguistics.

Bender, Emily M., Drellishak, Scott, Fokkens, Antske, Poulson, Laurie and Saleem, Safiyyah. 2010b. Grammar Customization. *Research on Language & Computation* 8(1), 23–72.

Bender, Emily M. and Flickinger, Dan. 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)*.

Bender, Emily M., Flickinger, Dan and Oepen, Stephan. 2002. The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars. In John Carroll, Nelleke Oostdijk and Richard Sutcliffe (eds.), *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pages 8–14.

Bender, Emily M., Poulson, Laurie, Drellishak, Scott and Evans, Chris. 2007. Validation and Regression Testing for a Cross-linguistic Grammar Resource. In *ACL 2007 Workshop on Deep Linguistic Processing*, pages 136–143, Association for Computational Linguistics.

Bresnan, Joan and Mchombo, Sam A. 1995. The Lexical Integrity Principle: Evidence from Bantu. *Natural Language & Linguistic Theory* 13(2), 181–254.

Budd, Peter. 2010. Negation in Bierebo and the Other Languages of Epi, Central Vanuatu. *Oceanic Linguistics* 49(2), 511–542.

Burquest, Donald A. 1973. *A Grammar of Angas*. Ph. D.thesis, University of California, Los Angeles.

Camargo, Eliane. 2002. Cashinahua Personal Pronouns in Grammatical Relations. *Indigenous Languages of Lating America* 3, 149–168.

Copestake, Ann. 2002. *Implementing Typed Feature Structure Grammars. 2002*. CSLI Publications.

Copestake, Ann, Flickinger, Dan, Pollard, Carl and Sag, Ivan A. 2005. Minimal Recursion Semantics: An Introduction. *Research on Language & Computation* 3(4), 281–332.

Crowgey, Joshua. 2012a. An a priori Typology of Sentential Negation from an HPSG Perspective. In Stefan Müller (ed.), *Proceedings of the 19th International Conference on Head-Driven Phrase Structure Grammar, Chungnam National University Daejeon*, pages 107–122.

Crowgey, Joshua. 2012b. What's a Morpheme: The Relationship of the Dravidian Zero Negative and Typological Universals to and Implented, Cross-Linguistic, Grammar-Engineering System. Presented the 19th International Conference on Head-Driven Phrase Structure Grammar, Chungnam National University Daejeon.

Crowgey, Joshua and Bender, Emily M. 2011. Analyzing Interacting Phenomena: Word Order and Negation in Basque. In Stefan Müller (ed.), *Proceedings of the HPSG11 Conference*, CSLI Publications.

Crysmann, Berthold. 2010. Discontinuous Negation in Hausa. In Stefan Müller (ed.), *Proceedings of the HPSG10 Conference*, CSLI Publications.

Dahl, Östen. 1979. Typology of Sentence Negation. *Linguistics* 17(1-2), 79–106.

De Angulo, Jaime and Freeland, L.S. 1930. The Achumawi Language. *International Journal of American Linguistics* 6(2), 77–120.

de Swart, Henriëtte and Sag, Ivan A. 2002. Negation and Negative Concord in Romance. *Linguistics and Philosophy* 25(4), 373–417.

Donaldson, Bruce C. 1993. *A Grammar of Afrikaans*, volume 8 of *Mouton Grammar Library*. Walter de Gruyter.

Drellishak, Scott. 2009. *Widespread but not universal: Improving the typological coverage of the Grammar Matrix*. Ph. D.thesis, University of Washington.

Dryer, Matthew S. 2005. Negative morphemes. *The World Atlas of Language Structures* pages 454–457.

Dryer, Matthew S. 2009. Verb-Object-Negative order in Central Africa. *Negation patterns in West African languages and beyond* 87, 307.

Flickinger, Dan. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering* 6(1), 15–28.

Flickinger, Dan and Bender, Emily M. 2003. Compositional Semantics in a Multilingual Grammar Resource. In Emily M. Bender, Dan Flickinger, Frederik Fouvry and Melanie Siegel (eds.), *Proceedings of the Workshop on Ideas and Strategies for Multilingual Grammar Devel, ESSLLI 2003opment*, pages 33–42.

Fokkens, Antske. 2010. Documentation for the Grammar Matrix word order library. Technical Report, Saarland University.

Ganong, Tina W. 1998. *Features of Baga Morphology, Syntax, and Narrative Discourse*. Ph. D.thesis, University of Texas at Arlington.

Goodman, Michael W. and Bender, Emily M. 2010. What's in a Word? Refining the Morphotactic Infrastructure in the Lingo Grammar Matrix Customization System. Presented at the Workshop on Morphology and Formal Grammar, Paris.

Hill, Jane H. 2005. *A Grammar of Cupeño*, volume 136 of *Linguistics*. University of California Press.

Hinrichs, Erhard and Nakazawa, Tsuneko. 1990. Subcategorization and VP Structure in German. In Shaun Hughes and Joe Salmons (eds.), *Proc. 3rd Symposium on Germanic Linguistics*, Benjamins.

Holton, Jim. 2004. *Chinook Jargon: The hidden language of the Pacific Northwest*. Wawa Press.

Hopcroft, John E., Motwani, Rajeev and Ullman, Jeffrey D. 2006. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley.

Horn, Laurence R. 2010. *The Expression of Negation*, volume 4 of *The Expression of Cognitive Categories*. Mouton de Gruyter.

Jespersen, Otto. 1924. *The Philosophy of Grammar*. Allen and Unwin.

Josephs, Lewis S. 1975. *Palauan Reference Grammar*, volume 229. University Press of Hawaii.

Jurafsky, D., Martin, J.H., Kehler, A., Vander Linden, K. and Ward, N. 2000. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, volume 163. MIT Press.

Kim, Jong Bok. 2000. *The Grammar of Negation: a constraint-based approach*. CSLI Publications Stanford.

Kim, Jong Bok and Sag, Ivan A. 2002. Negation without Head-Movement. *Natural Language & Linguistic Theory* 20(2), 339–412.

Lucas, Christopher and Lash, Elliott. 2008. Contact as catalyst: the case for Coptic influence in the development of Arabic negation. *Manuscript Cambridge University* .

Maslova, Elena. 2003. *A Grammar of Kolyma Yukaghir*, volume 27 of *Mouton Grammar Library*. De Gruyter Mouton.

Master, Alfred. 1946. The Zero Negative in Dravidian. *Transactions of the Philological Society* 45(1), 137–155.

Meier, Paul, Meier, Ingeborg and Bendor-Samuel, John. 1975. *A grammar of Izi, an Igbo language*. Summer Institute of Linguistics.

Meyer-Bahlburg, Hilke. 1972. *Studien zur Morphologie und Syntax des Musgu*. Helmut Buske Verlag.

Miestamo, Matti. 2005. *Standard Negation: The Negation of Declarative Verbal Main Clauses in a Typological Perspective*. Walter de Gruyter.

Miestamo, Matti. 2010. Negatives without Negators. *Rethinking Universals: How Rarities Affect Linguistic Theory* 45, 169.

Montag, Richard. 1973. La estructura semántica de las relaciones entre frases verbales en cashinahua. *Estudios Panos* 2, 107–159.

Montag, Richard. 2005. Participant referencing in Cashinahua. *SIL Electronic Working Papers* 13.

Montag, Susan. 2004. Lições para a aprendizagem da língua Kaxinawá. *Dados Etno-Lingüísticos 59* .

Moran, Steve and Wright, Richard. 2009. Phonetics information base and lexicon (PHOIBLE). *Online: http://phoible. org* .

Oosthuizen, Johan. 1998. The Final nie in Afrikaans Negative Sentences. *Stellenbosch Papers in Linguistics* 31(6), 92.

Pollard, Carl J. and Sag, Ivan A. 1994. *Head-driven Phrase Structure Grammar*. University of Chicago Press.

Pollock, Jean-Yves. 1989. Verb movement, Universal Grammar, and the structure of IP. *Linguistic Inquiry* 20(3), 365–424.

Sag, Ivan A., Wasow, Thomas and Bender, Emily M. 2003. *Syntactic Theory: A formal introduction*. CSLI Publications.

Saleem, Safiyyah. 2010. *Argument Optionality: A New Library for the Grammar Matrix Customization System*. Ph. D.thesis, University of Washington.

Shieber, Stuart M. 1985. Evidence against the Context-Freeness of Natural Language. *Linguistics and Philosophy* 8(3), 333–343.

Sulkala, Helena and Karjalainen, Merja. 1992. *Finnish*. Psychology Press.

Tucker, Archibald N. and Bryan, Margaret A. 1966. *Linguistic Analyses: the non-Bantu languages of North-Eastern Africa*. Oxford University Press.

Zwicky, Arnold M. 1985. Heads. *Journal of linguistics* 21(01), 1–29.

Appendix A

CHOICES AND TEST-SUITES FOR HELD-OUT LANGUAGES

Below I include the test-suites annotated with parse results and choices files for each of the languages in the evaluation experiment described in Chapter 6.2. To save space, the choices files are abbreviated in that sections names for empty sections are supressed. The parse results show (from the left on each line) the item number, the annotated string, the number of parses found and the number of edges considered by the parsing algorithm during the search.

A.1 Baga Sitemu [bsp]

```
version=26

section=general
language=Baga Sitemu
iso-code=bsp
punctuation-chars=keep-all

section=word-order
word-order=svo
has-dets=no
has-aux=no

section=number
  number1_name=sg
  number2_name=pl

section=person
person=1-2-3
first-person=none

section=gender
  gender1_name=sing-class
  gender2_name=pl-class
  gender3_name=1+2
  gender4_name=1
    gender4_supertype1_name=sing-class
    gender4_supertype2_name=1+2
  gender5_name=2
    gender5_supertype1_name=pl-class
    gender5_supertype2_name=1+2
  gender6_name=5+6
  gender7_name=5
    gender7_supertype1_name=5+6
    gender7_supertype2_name=sing-class

section=tense-aspect-mood
subjind=on

section=sentential-negation
neg-exp=1
infl-neg=on

section=arg-opt
subj-drop=subj-drop-all
subj-mark-drop=subj-mark-drop-req
subj-mark-no-drop=subj-mark-no-drop-req
subj-con=subj-con-always
obj-drop=obj-drop-all
obj-mark-drop=obj-mark-drop-req
obj-mark-no-drop=obj-mark-no-drop-req

section=lexicon
  noun1_name=1+2-nstem
    noun1_feat1_name=gender
    noun1_feat1_value=1+2
  noun1_det=imp
    noun1_stem1_orth=bɛ
    noun1_stem1_pred=_b_n_rel
  noun2_name=5+6-nstem
    noun2_feat1_name=gender
    noun2_feat1_value=5+6
  noun2_det=imp
    noun2_stem1_orth=ɔkɔ
    noun2_stem1_pred=_ɔkɔ_n_rel
  verb1_name=intrans
  verb1_valence=intrans
    verb1_stem1_orth=tam
    verb1_stem1_pred=_tam_v_rel
```

```
    verb1_stem2_orth=cuca
    verb1_stem2_pred=_cuca_v_rel
  verb2_name=trans
  verb2_valence=trans
    verb2_stem1_orth=nɨŋk
    verb2_stem1_pred=_nɨŋk_v_rel

section=morphology
  noun-pc1_name=nc
  noun-pc1_obligatory=on
  noun-pc1_order=prefix
  noun-pc1_inputs=noun
    noun-pc1_lrt1_name=nc1
      noun-pc1_lrt1_feat1_name=gender
      noun-pc1_lrt1_feat1_value=1
      noun-pc1_lrt1_feat2_name=number
      noun-pc1_lrt1_feat2_value=sg
      noun-pc1_lrt1_lri1_inflecting=yes
      noun-pc1_lrt1_lri1_orth=wɨ-
    noun-pc1_lrt2_name=nc2
      noun-pc1_lrt2_feat1_name=gender
      noun-pc1_lrt2_feat1_value=2
      noun-pc1_lrt2_feat2_name=number
      noun-pc1_lrt2_feat2_value=pl
      noun-pc1_lrt2_lri1_orth=a-
    noun-pc1_lrt3_name=nc5
      noun-pc1_lrt3_feat1_name=gender
      noun-pc1_lrt3_feat1_value=5
      noun-pc1_lrt3_feat2_name=number
      noun-pc1_lrt3_feat2_value=sg
      noun-pc1_lrt3_lri1_inflecting=yes
      noun-pc1_lrt3_lri1_orth=tat-
  verb-pc1_name=nasal
  verb-pc1_obligatory=on
  verb-pc1_order=prefix
  verb-pc1_inputs=verb
    verb-pc1_lrt1_name=nrule
      verb-pc1_lrt1_lri1_inflecting=yes
      verb-pc1_lrt1_lri1_orth=n-
    verb-pc1_lrt2_name=null_nrule
      verb-pc1_lrt2_lri1_inflecting=no
  verb-pc2_name=subj_agr
  verb-pc2_obligatory=on
  verb-pc2_order=prefix
  verb-pc2_inputs=verb-pc1, verb-pc6
    verb-pc2_lrt1_name=1sg
    verb-pc2_lrt1_supertypes=
             verb-pc2_lrt8, verb-pc2_lrt9
      verb-pc2_lrt1_lri1_orth=i-
    verb-pc2_lrt2_name=2sg
    verb-pc2_lrt2_supertypes=
             verb-pc2_lrt8, verb-pc2_lrt10
      verb-pc2_lrt2_lri1_inflecting=yes
      verb-pc2_lrt2_lri1_orth=mɨ-
    verb-pc2_lrt3_name=3sg-1
    verb-pc2_lrt3_supertypes=
             verb-pc2_lrt8, verb-pc2_lrt11
      verb-pc2_lrt3_feat1_name=gender
      verb-pc2_lrt3_feat1_value=1
      verb-pc2_lrt3_feat1_head=subj
      verb-pc2_lrt3_lri1_inflecting=yes
      verb-pc2_lrt3_lri1_orth=ɛ-
      verb-pc2_lrt3_lri2_inflecting=yes
      verb-pc2_lrt3_lri2_orth=e-
      verb-pc2_lrt3_lri3_inflecting=yes
      verb-pc2_lrt3_lri3_orth=ɔ-
    verb-pc2_lrt4_name=1pl
    verb-pc2_lrt4_supertypes=
             verb-pc2_lrt7, verb-pc2_lrt9
      verb-pc2_lrt4_lri1_inflecting=yes
      verb-pc2_lrt4_lri1_orth=sɨ-
    verb-pc2_lrt5_name=2pl
    verb-pc2_lrt5_supertypes=
             verb-pc2_lrt7, verb-pc2_lrt10
      verb-pc2_lrt5_lri1_inflecting=yes
      verb-pc2_lrt5_lri1_orth=nɨ-
    verb-pc2_lrt6_name=3pl-2
    verb-pc2_lrt6_supertypes=
             verb-pc2_lrt7, verb-pc2_lrt11
      verb-pc2_lrt6_feat1_name=gender
      verb-pc2_lrt6_feat1_value=2
      verb-pc2_lrt6_feat1_head=subj
      verb-pc2_lrt6_lri1_inflecting=yes
      verb-pc2_lrt6_lri1_orth=ŋa-
    verb-pc2_lrt7_name=pl
      verb-pc2_lrt7_feat1_name=number
      verb-pc2_lrt7_feat1_value=pl
      verb-pc2_lrt7_feat1_head=subj
    verb-pc2_lrt8_name=sg
      verb-pc2_lrt8_feat1_name=number
      verb-pc2_lrt8_feat1_value=sg
      verb-pc2_lrt8_feat1_head=subj
    verb-pc2_lrt9_name=1
      verb-pc2_lrt9_feat1_name=person
      verb-pc2_lrt9_feat1_value=1st
      verb-pc2_lrt9_feat1_head=subj
    verb-pc2_lrt10_name=2
      verb-pc2_lrt10_feat1_name=person
      verb-pc2_lrt10_feat1_value=2nd
      verb-pc2_lrt10_feat1_head=subj
    verb-pc2_lrt11_name=3
      verb-pc2_lrt11_feat1_name=person
      verb-pc2_lrt11_feat1_value=3rd
      verb-pc2_lrt11_feat1_head=subj
    verb-pc2_lrt12_name=3sg-5
    verb-pc2_lrt12_supertypes=
             verb-pc2_lrt8, verb-pc2_lrt11
      verb-pc2_lrt12_feat1_name=gender
      verb-pc2_lrt12_feat1_value=5
      verb-pc2_lrt12_feat1_head=subj
      verb-pc2_lrt12_lri1_inflecting=yes
      verb-pc2_lrt12_lri1_orth=tɔ-
  verb-pc3_name=suff1
  verb-pc3_order=suffix
  verb-pc3_inputs=verb-pc2
    verb-pc3_lrt1_name=rcpr
      verb-pc3_lrt1_lri1_inflecting=yes
      verb-pc3_lrt1_lri1_orth=-ɨnɛ
  verb-pc4_name=neg
  verb-pc4_order=suffix
  verb-pc4_inputs=verb-pc2, verb-pc3
    verb-pc4_lrt1_name=negsuff
      verb-pc4_lrt1_feat1_name=negation
      verb-pc4_lrt1_feat1_value=plus
      verb-pc4_lrt1_feat1_head=verb
      verb-pc4_lrt1_feat2_name=mood
```

```
    verb-pc4_lrt1_feat2_value=indicative
    verb-pc4_lrt1_feat2_head=verb
    verb-pc4_lrt1_lri1_inflecting=yes
    verb-pc4_lrt1_lri1_orth=-fɛ
  verb-pc5_name=o_agr
  verb-pc5_order=suffix
  verb-pc5_inputs=verb-pc3, verb-pc4
    verb-pc5_lrt1_name=3sO-1+2
      verb-pc5_lrt1_feat1_name=person
      verb-pc5_lrt1_feat1_value=3rd
      verb-pc5_lrt1_feat1_head=obj
      verb-pc5_lrt1_feat2_name=number
      verb-pc5_lrt1_feat2_value=sg
      verb-pc5_lrt1_feat2_head=obj
      verb-pc5_lrt1_feat3_name=gender
      verb-pc5_lrt1_feat3_value=1+2
      verb-pc5_lrt1_feat3_head=obj
      verb-pc5_lrt1_require1_others=tverb
      verb-pc5_lrt1_lri1_inflecting=yes
      verb-pc5_lrt1_lri1_orth=-kɔ
      verb-pc5_lrt1_lri2_inflecting=yes
      verb-pc5_lrt1_lri2_orth=-ki
    verb-pc5_lrt2_name=3s0-5
      verb-pc5_lrt2_feat1_name=person
      verb-pc5_lrt2_feat1_value=3rd
      verb-pc5_lrt2_feat1_head=obj
      verb-pc5_lrt2_feat2_name=number
      verb-pc5_lrt2_feat2_value=sg
      verb-pc5_lrt2_feat2_head=obj
      verb-pc5_lrt2_feat3_name=gender
      verb-pc5_lrt2_feat3_value=5
      verb-pc5_lrt2_feat3_head=obj
      verb-pc5_lrt2_lri1_inflecting=yes
      verb-pc5_lrt2_lri1_orth=-ti
  verb-pc6_name=irrneg
  verb-pc6_order=prefix
  verb-pc6_inputs=verb-pc1
    verb-pc6_lrt1_name=irrneg-pref
      verb-pc6_lrt1_feat1_name=negation
      verb-pc6_lrt1_feat1_value=plus
      verb-pc6_lrt1_feat1_head=verb
      verb-pc6_lrt1_feat2_name=mood
      verb-pc6_lrt1_feat2_value=subjunctive
      verb-pc6_lrt1_feat2_head=verb
      verb-pc6_lrt1_lri1_inflecting=yes
      verb-pc6_lrt1_lri1_orth=fɔ-
```

The Baga test-suite tests the two negative markers (1), (2) and their inability to cooccur (3). It checks that noun inflection and agreement are working with negatives (4–7) and that incorrect ordersof attachment are not licensed (8–11)

```
1 ɛ-nɨŋk-fɛ-kɔ 1 12
2 mɨ-fɔ-nɨŋk-ki 1 12
3 *mɨ-fɔ-nɨŋk-fɛ-ki 0 1
4 tat-ɔkɔ tɔ-n-cuca-fɛ 1 13
5 *tat-ɔkɔ ɛ-n-cuca-fɛ 0 8
6 *wɨ-ɔkɔ ɛ-n-cuca-fɛ 0 8
7 ɛ-n-tam-ɨnɛ-fɛ 1 8
8 *ɛ-n-tam-fɛ-ɨnɛ 0 1
9 *-fɛɛ-n-tam-ɨnɛ 0 1
10 *ɛ-n--fɛtam-ɨnɛ 0 1
11 *ɛ--fɛn-tam-ɨnɛ 0 1
```

A.2 Kashinawa [cbs]

```
version=26

section=general
language=Cashinahua
iso-code=cbs

section=word-order
word-order=sov
has-dets=no
has-aux=no

section=number
  number1_name=sg
  number2_name=pl

section=person
person=1-2-3
first-person=none

section=case
```

```
case-marking=split-n
split-n-nom-case-name=nom
split-n-acc-case-name=acc
split-n-erg-case-name=erg
split-n-abs-case-name=abs

section=tense-aspect-mood
  mood1_name=decl
    mood1_supertype1_name=mood

section=sentential-negation
neg-exp=1
infl-neg=on

section=arg-opt
subj-drop=subj-drop-all
subj-mark-drop=subj-mark-drop-opt
subj-mark-no-drop=subj-mark-no-drop-opt
subj-con=subj-con-always
obj-drop=obj-drop-all
obj-mark-drop=obj-mark-drop-opt
obj-mark-no-drop=obj-mark-no-drop-opt

section=lexicon
  noun1_name=pronoun
  noun1_det=imp
  noun2_name=subj-pron
  noun2_supertypes=noun1
    noun2_feat1_name=case
    noun2_feat1_value=nom
  noun2_det=imp
  noun3_name=1sg-subj
  noun3_supertypes=noun2, noun12
    noun3_feat2_name=number
    noun3_feat2_value=sg
  noun3_det=imp
    noun3_stem1_orth=ẽ
    noun3_stem1_pred=_ẽ_n_rel
  noun4_name=2sg-subj
  noun4_supertypes=noun2, noun13
    noun4_feat2_name=number
    noun4_feat2_value=sg
  noun4_det=imp
    noun4_stem1_orth=mĩ
    noun4_stem1_pred=_mĩ_n_rel
  noun5_name=3sg-subj
  noun5_supertypes=noun2, noun14
    noun5_feat2_name=number
    noun5_feat2_value=sg
  noun5_det=imp
    noun5_stem1_orth=hatũ
    noun5_stem1_pred=_hatũ_n_rel
  noun6_name=1pl-subj
  noun6_supertypes=noun2, noun12
    noun6_feat1_name=number
    noun6_feat1_value=pl
  noun6_det=imp
    noun6_stem1_orth=nũ
    noun6_stem1_pred=_nũ_n_rel
  noun7_name=2pl-subj
  noun7_supertypes=noun2, noun13
    noun7_feat1_name=number
    noun7_feat1_value=pl
noun7_det=imp
  noun7_stem1_orth=mã
  noun7_stem1_pred=_mã_n_rel
noun8_name=3pl-subj
noun8_supertypes=noun2, noun14
  noun8_feat1_name=number
  noun8_feat1_value=pl
noun8_det=imp
  noun8_stem1_orth=habũ
  noun8_stem1_pred=_habũ_n_rel
noun9_name=obj-pron
noun9_supertypes=noun1
  noun9_feat1_name=case
  noun9_feat1_value=acc
noun9_det=imp
noun10_name=1sg-obj
noun10_supertypes=noun9, noun12
  noun10_feat1_name=number
  noun10_feat1_value=sg
noun10_det=imp
  noun10_stem1_orth=ea
  noun10_stem1_pred=_ea _n_rel
noun11_name=2sg-obj
noun11_supertypes=noun9, noun13
  noun11_feat1_name=number
  noun11_feat1_value=sg
noun11_det=imp
  noun11_stem1_orth=mia
  noun11_stem1_pred=_mia_n_rel
noun12_name=1per
  noun12_feat1_name=person
  noun12_feat1_value=1st
noun12_det=opt
noun13_name=2per
  noun13_feat1_name=person
  noun13_feat1_value=2nd
noun13_det=opt
noun14_name=3per
  noun14_feat1_name=person
  noun14_feat1_value=3rd
noun14_det=opt
noun15_name=3sg-obj
noun15_supertypes=noun9, noun14
  noun15_feat1_name=number
  noun15_feat1_value=sg
noun15_det=imp
  noun15_stem1_orth=hatu
  noun15_stem1_pred=_hatu_n_rel
noun16_name=1pl-obj
noun16_supertypes=noun9, noun12
  noun16_feat1_name=number
  noun16_feat1_value=pl
noun16_det=imp
  noun16_stem1_orth=nuku
  noun16_stem1_pred=_nuku_n_rel
noun17_name=2pl-obj
noun17_supertypes=noun9, noun13
  noun17_feat1_name=number
  noun17_feat1_value=pl
noun17_det=imp
  noun17_stem1_orth=matu
  noun17_stem1_pred=_matu_n_rel
noun18_name=3pl-obj
```

```
  noun18_supertypes=noun9, noun14
    noun18_feat1_name=number
    noun18_feat1_value=pl
  noun18_det=imp
    noun18_stem1_orth=habu
    noun18_stem1_pred=_habu_n_rel
  verb1_name=intrans-sg
    verb1_feat1_name=number
    verb1_feat1_value=sg
    verb1_feat1_head=subj
  verb1_valence=intrans
    verb1_stem1_orth=hu
    verb1_stem1_pred=_hu_v_rel
    verb1_stem2_orth=ka
    verb1_stem2_pred=_ka_v_rel
  verb2_name=intrans-pl
    verb2_feat1_name=number
    verb2_feat1_value=pl
    verb2_feat1_head=subj
  verb2_valence=intrans
    verb2_stem1_orth=bekã
    verb2_stem1_pred=_bekã_v_rel
    verb2_stem2_orth=bukã
    verb2_stem2_pred=_bukã_v_rel
  verb3_name=trans
  verb3_valence=a_case-o_case
    verb3_stem1_orth=pi
    verb3_stem1_pred=_pi_v_rel

section=morphology
  verb-pc1_name=nega
  verb-pc1_order=suffix
  verb-pc1_inputs=verb
    verb-pc1_require1_others=verb-pc3
    verb-pc1_lrt1_name=neg1suff
      verb-pc1_lrt1_lri1_inflecting=yes
      verb-pc1_lrt1_lri1_orth=a
 verb-pc2_name=plural
 verb-pc2_order=suffix
 verb-pc2_inputs=verb, verb-pc1
   verb-pc2_lrt1_name=pluralsuff
     verb-pc2_lrt1_feat1_name=number
     verb-pc2_lrt1_feat1_value=pl
     verb-pc2_lrt1_feat1_head=subj
     verb-pc2_lrt1_lri1_inflecting=yes
     verb-pc2_lrt1_lri1_orth=bu
 verb-pc3_name=negb
 verb-pc3_order=suffix
verb-pc3_inputs=verb-pc1_lrt1, verb-pc2_lrt1
   verb-pc3_require1_others=verb-pc1
   verb-pc3_lrt1_name=neg2suff
     verb-pc3_lrt1_feat1_name=negation
     verb-pc3_lrt1_feat1_value=plus
     verb-pc3_lrt1_feat1_head=verb
     verb-pc3_lrt1_lri1_inflecting=yes
     verb-pc3_lrt1_lri1_orth=ma
 verb-pc4_name=mood
 verb-pc4_obligatory=on
 verb-pc4_order=suffix
 verb-pc4_inputs=verb, verb-pc2, verb-pc3
   verb-pc4_lrt1_name=decl
     verb-pc4_lrt1_feat1_name=mood
     verb-pc4_lrt1_feat1_value=decl
     verb-pc4_lrt1_feat1_head=verb
     verb-pc4_lrt1_lri1_inflecting=yes
     verb-pc4_lrt1_lri1_orth=ki
```

The Kashinawa test-suite tests agreement between plurality of the subject and negative inflection and ensures morphology appears in the correct order.

```
1 Ẽ piamaki  1 13
2 Ẽ piki 1 11
3 piabumaki  1 9
4 *pibuamaki 0 1
5 *piamabuki 0 1
6 *pikiabuma 0 1
7 *piaki 0 5
8 *piabuki 0 6
9 *pibumaki 0 4
10 pibuki  1 7
11 *Ẽ pibuki 0 11
12 *Ẽ piabumaki 0 13
```

A.3 Chinook Jargon [chn]

```
version=26

section=general
language=Chinook
iso-code=chh
punctuation-chars=discard-all

section=word-order
word-order=svo
has-dets=yes
noun-det-order=det-noun
has-aux=no

section=number
  number1_name=sg
  number2_name=pl

section=person
person=1-2-3
first-person=none

section=case
case-marking=none
nom-acc-nom-case-name=nom
nom-acc-acc-case-name=acc

section=tense-aspect-mood
  mood1_name=decl
    mood1_supertype1_name=mood

section=sentential-negation
neg-exp=1
adv-neg=on
neg-mod=s
neg-order=before
neg-adv-orth=wik

section=arg-opt
subj-drop=subj-drop-all
subj-mark-drop=subj-mark-drop-opt
subj-mark-no-drop=subj-mark-no-drop-opt
subj-con=subj-con-always
obj-drop=obj-drop-all
obj-mark-drop=obj-mark-drop-opt
obj-mark-no-drop=obj-mark-no-drop-opt

section=lexicon
  noun1_name=pronoun
  noun1_det=imp
  noun3_name=1sg
  noun3_supertypes=noun9, noun1
    noun3_feat2_name=number
    noun3_feat2_value=sg
    noun3_stem1_orth=nayka
    noun3_stem1_pred=_1SG_pron_n_rel
  noun4_name=2sg
  noun4_supertypes=noun10, noun1
    noun4_feat2_name=number
    noun4_feat2_value=sg
    noun4_stem1_orth=mayka
    noun4_stem1_pred=_2SG_pron_n_rel
noun5_name=3sg
noun5_supertypes=noun11, noun1
  noun5_feat2_name=number
  noun5_feat2_value=sg
  noun5_stem1_orth=yaka
  noun5_stem1_pred=_3SG_pron_n_rel
noun6_name=1pl
noun6_supertypes=noun9, noun1
  noun6_feat1_name=number
  noun6_feat1_value=pl
  noun6_stem1_orth=nêsayka
  noun6_stem1_pred=_1PL_pron_n_rel
noun7_name=2pl
noun7_supertypes=noun10, noun1
  noun7_feat1_name=number
  noun7_feat1_value=pl
  noun7_stem1_orth=mêsayka
  noun7_stem1_pred=_2PL_pron_n_rel
noun8_name=3pl
noun8_supertypes=noun11, noun1
  noun8_feat1_name=number
  noun8_feat1_value=pl
  noun8_stem1_orth=klaska
  noun8_stem1_pred=_3PL_pron_n_rel
noun9_name=1per
  noun9_feat1_name=person
  noun9_feat1_value=1st
noun9_det=opt
noun10_name=2per
  noun10_feat1_name=person
  noun10_feat1_value=2nd
noun10_det=opt
noun11_name=3per
  noun11_feat1_name=person
  noun11_feat1_value=3rd
noun11_det=opt
noun12_name=common
noun12_supertypes=noun11
noun12_det=opt
  noun12_stem1_orth=man
  noun12_stem1_pred=_man_n_rel
  noun12_stem2_orth=libal
  noun12_stem2_pred=_libal_n_rel
  noun12_stem3_orth=ikta
  noun12_stem3_pred=_ikta_n_rel
verb1_name=intrans
verb1_valence=intrans
  verb1_stem1_orth=nanich
  verb1_stem1_pred=_nanich_v_1_rel
verb2_name=trans
verb2_valence=trans
  verb2_stem1_orth=nanich
  verb2_stem1_pred=_nanich_v_2_rel
det1_name=specific
  det1_stem1_orth=ukuk
  det1_stem1_pred=_ukuk_q_rel
det2_name=numeral
  det2_stem1_orth=ikt
  det2_stem1_pred=_ikt_q_rel
```

The Chinook Jargon test-suite checks for the proper placement of the negative adverb

for both intransitive and transitive clauses. The two parses for (1) and (2) are due to the fact that the grammar includes both an intransitive and transitive sense for *nanich* ('I see').

```
1 nayka nanich 2 13
2 wik nayka nanich 2 18
3 *nayka wik nanich 0 16
4 *nayka nanich wik  0 16
5 nayka nanich libal 1 20
6 wik nayka nanich libal 1 26
7 *nayka wik nanich libal 0 23
8 *nayka nanich wik libal 0 20
9 hawkwêtl nayka nanich 0 14
```

A.4 Cupeño [cup]

```
version=26

section=general
language=Cupeño
iso-code=cup
punctuation-chars=keep-list
punctuation-chars-list=-'=

section=word-order
word-order=sov
has-dets=no
has-aux=yes
aux-comp-order=before
aux-comp=vp
multiple-aux=no

section=number
  number1_name=sg
  number2_name=pl

section=person
person=1-2-3
first-person=none

section=tense-aspect-mood
tense-definition=choose
past=on
present=on
future=on
nonpast=on
nonfuture=on
  mood1_name=irr
    mood1_supertype1_name=mood
  mood2_name=rep
    mood2_supertype1_name=mood

section=sentential-negation
neg-exp=1
adv-neg=on
neg-mod=s
neg-order=before
neg-adv-orth=qay

section=arg-opt
subj-drop=subj-drop-all
subj-mark-drop=subj-mark-drop-opt
subj-mark-no-drop=subj-mark-no-drop-opt
subj-con=subj-con-always
obj-drop=obj-drop-all
obj-mark-drop=obj-mark-drop-opt
obj-mark-no-drop=obj-mark-no-drop-opt

section=lexicon
  noun1_name=common
  noun1_det=imp
    noun1_stem1_orth='ye
    noun1_stem1_pred=_'ye_n_rel
  verb1_name=yax
  verb1_valence=intrans
    verb1_stem1_orth=muyaq
    verb1_stem1_pred=_muyaq_v_rel
    verb1_stem2_orth=hiwen
    verb1_stem2_pred=_hiwen_v_rel
  verb2_name=in
  verb2_valence=trans
  verb3_name=nil
  verb3_valence=trans
    verb3_stem1_orth=miyax
    verb3_stem1_pred=_miyax_v_rel
  aux1_name=clitic-complex
  aux1_sem=no-pred
  aux1_subj=np
    aux1_compfeature1_name=form
    aux1_compfeature1_value=nonfinite
    aux1_stem1_orth==

section=morphology
  noun-pc1_name=pre
  noun-pc1_order=prefix
  noun-pc1_inputs=noun1
```

```
  noun-pc1_lrt1_name=dup
    noun-pc1_lrt1_lri1_inflecting=yes
    noun-pc1_lrt1_lri1_orth=DUP-
noun-pc2_name=uto-abs
noun-pc2_order=suffix
noun-pc2_inputs=noun1, noun-pc1
  noun-pc2_lrt1_name=npn
    noun-pc2_lrt1_lri1_inflecting=yes
    noun-pc2_lrt1_lri1_orth=-t
noun-pc3_name=number
noun-pc3_order=suffix
noun-pc3_inputs=noun1, noun-pc1, noun-pc2
  noun-pc3_lrt1_name=plural
    noun-pc3_lrt1_feat1_name=number
    noun-pc3_lrt1_feat1_value=pl
    noun-pc3_lrt1_lri1_inflecting=yes
    noun-pc3_lrt1_lri1_orth=-im
verb-pc1_name=complex1
verb-pc1_obligatory=on
verb-pc1_order=suffix
verb-pc1_inputs=aux
  verb-pc1_lrt1_name=2plS
    verb-pc1_lrt1_feat1_name=person
    verb-pc1_lrt1_feat1_value=2nd
    verb-pc1_lrt1_feat1_head=subj
    verb-pc1_lrt1_feat2_name=number
    verb-pc1_lrt1_feat2_value=pl
    verb-pc1_lrt1_feat2_head=subj
    verb-pc1_lrt1_lri1_inflecting=yes
    verb-pc1_lrt1_lri1_orth=el
  verb-pc1_lrt2_name=repclit
    verb-pc1_lrt2_feat1_name=mood
    verb-pc1_lrt2_feat1_value=rep
    verb-pc1_lrt2_feat1_head=verb
    verb-pc1_lrt2_lri1_inflecting=yes
    verb-pc1_lrt2_lri1_orth=ku'ut
verb-pc2_name=complex2
verb-pc2_order=suffix
verb-pc2_inputs=verb-pc1
  verb-pc2_lrt1_name=irrmood
    verb-pc2_lrt1_feat1_name=mood
    verb-pc2_lrt1_feat1_value=irr
    verb-pc2_lrt1_feat1_head=verb
  verb-pc2_lrt1_forbid1_others=verb-pc1_lrt2
    verb-pc2_lrt1_lri1_inflecting=yes
    verb-pc2_lrt1_lri1_orth==pe
verb-pc3_name=yax1-PNS
verb-pc3_obligatory=on
verb-pc3_order=suffix
verb-pc3_inputs=verb1
  verb-pc3_lrt1_name=3sg
    verb-pc3_lrt1_feat1_name=number
    verb-pc3_lrt1_feat1_value=sg
    verb-pc3_lrt1_feat1_head=subj
    verb-pc3_lrt1_feat2_name=person
    verb-pc3_lrt1_feat2_value=3rd
    verb-pc3_lrt1_feat2_head=subj
    verb-pc3_lrt1_feat4_name=tense
    verb-pc3_lrt1_feat4_value=past
    verb-pc3_lrt1_feat4_head=verb
    verb-pc3_lrt1_lri1_inflecting=yes
    verb-pc3_lrt1_lri1_orth=-pe
  verb-pc3_lrt2_name=nonpastempty
    verb-pc3_lrt2_feat1_name=tense
    verb-pc3_lrt2_feat1_value=nonpast
    verb-pc3_lrt2_feat1_head=verb
    verb-pc3_lrt2_lri1_inflecting=no
verb-pc4_name=yaxtheme
verb-pc4_obligatory=on
verb-pc4_order=suffix
verb-pc4_inputs=verb-pc3
    verb-pc4_lrt1_lri1_inflecting=yes
    verb-pc4_lrt1_lri1_orth=-yax
verb-pc5_name=yaxclit
verb-pc5_order=suffix
verb-pc5_inputs=verb-pc4
  verb-pc5_lrt1_name=rep
    verb-pc5_lrt1_feat1_name=mood
    verb-pc5_lrt1_feat1_value=rep
    verb-pc5_lrt1_feat1_head=verb
    verb-pc5_lrt1_lri1_inflecting=yes
    verb-pc5_lrt1_lri1_orth==ku'ut
verb-pc6_name=nil1-PNS
verb-pc6_obligatory=on
verb-pc6_order=prefix
verb-pc6_inputs=verb3
  verb-pc6_lrt1_name=3plsubj
    verb-pc6_lrt1_feat1_name=number
    verb-pc6_lrt1_feat1_value=pl
    verb-pc6_lrt1_feat1_head=subj
    verb-pc6_lrt1_feat2_name=person
    verb-pc6_lrt1_feat2_value=3rd
    verb-pc6_lrt1_feat2_head=subj
    verb-pc6_lrt1_feat3_name=tense
    verb-pc6_lrt1_feat3_value=past
    verb-pc6_lrt1_feat3_head=verb
    verb-pc6_lrt1_lri1_inflecting=yes
    verb-pc6_lrt1_lri1_orth=pe'-
  verb-pc6_lrt2_name=nilnonpast
    verb-pc6_lrt2_feat1_name=tense
    verb-pc6_lrt2_feat1_value=nonpast
    verb-pc6_lrt2_feat1_head=verb
    verb-pc6_lrt2_lri1_inflecting=no
verb-pc7_name=tam-num
verb-pc7_order=suffix
verb-pc7_inputs=verb-pc6
  verb-pc7_lrt1_name=pipl
    verb-pc7_lrt1_feat1_name=tense
    verb-pc7_lrt1_feat1_value=past
    verb-pc7_lrt1_feat1_head=verb
    verb-pc7_lrt1_feat2_name=number
    verb-pc7_lrt1_feat2_value=pl
    verb-pc7_lrt1_feat2_head=subj
    verb-pc7_lrt1_lri1_inflecting=yes
    verb-pc7_lrt1_lri1_orth=-wen
```

The test-suite for Cupeño ensures that the negative marker can only occur initially. The two parse results on item (2) are due to the fact that the that either argument can be

dropped.

```
1 qay =el=pe muyaq-yax 1 18
2 qay DUP-'ye-t-im pe'-miyax-wen 2 22
3 qay =ku'ut hiwen-pe-yax=ku'ut 1 17
4 *=el=pe qay muyaq-yax 0 16
5 *=el=pe muyaq-yax qay  0 17
6 *DUP-'ye-t-im qay pe'-miyax-wen 0 18
7 *DUP-'ye-t-im pe'-miyax-wen qay  0 20
8 *=ku'ut qay hiwen-pe-yax=ku'ut 0 15
9 *=ku'ut hiwen-pe-yax=ku'ut qay 0 16
```

A.5 Palauan [pau]

```
version=26

section=general
language=abstract-pau-neg

section=word-order
word-order=svo
has-dets=yes
noun-det-order=det-noun
has-aux=yes
aux-comp-order=before
aux-comp=v
multiple-aux=no

section=case
case-marking=none

section=tense-aspect-mood
  mood1_name=ind
    mood1_supertype1_name=mood
  mood2_name=hyp
    mood2_supertype1_name=mood

section=sentential-negation
neg-exp=1
neg-aux-index=1

section=lexicon
  noun1_name=common
  noun1_det=obl
    noun1_stem1_orth=child
    noun1_stem1_pred=_child_n_rel
  verb1_name=intrans
    verb1_valence=intrans
      verb1_stem1_orth=sick
      verb1_stem1_pred=_sick_v_rel
    aux1_name=neg
    aux1_sem=add-pred
    aux1_subj=np
      aux1_compfeature1_name=form
      aux1_compfeature1_value=nonfinite
      aux1_compfeature2_name=mood
      aux1_compfeature2_value=hyp
      aux1_stem1_orth=negaux
      aux1_stem1_pred=neg_rel
      det1_stem1_orth=det
      det1_stem1_pred=_det_q_rel

section=morphology
  verb-pc1_name=mood
  verb-pc1_obligatory=on
  verb-pc1_order=prefix
  verb-pc1_inputs=iverb, tverb
    verb-pc1_lrt1_name=hyp
      verb-pc1_lrt1_feat1_name=mood
      verb-pc1_lrt1_feat1_value=hyp
      verb-pc1_lrt1_feat1_head=verb
      verb-pc1_lrt1_lri1_inflecting=yes
      verb-pc1_lrt1_lri1_orth=hyp-
    verb-pc1_lrt2_name=ind
      verb-pc1_lrt2_feat1_name=mood
      verb-pc1_lrt2_feat1_value=ind
      verb-pc1_lrt2_feat1_head=verb
      verb-pc1_lrt2_lri1_inflecting=yes
      verb-pc1_lrt2_lri1_orth=ind-
```

The abstract Paluan test-suite tests that negative auxiliaeis can only appear when the verb is marked for hypothetical mood.

```
1 det child negaux hyp-sick 1 16
2 det child ind-sick 1 12
```

```
3 det child hyp-sick 1 12
4 *det child negaux ind-sick 0 14
```