A Parametric Implementation of Valence-changing Morphology in the LinGO Grammar Matrix

Christian M. Curtis

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

Master of Science

University of Washington

2018

Reading Committee:

Emily M. Bender, Chair

Sanghoun Song

Program Authorized to Offer Degree:
Linguistics

University of Washington

Abstract

A Parametric Implementation of Valence-changing Morphology in the LinGO Grammar Matrix

Christian M. Curtis

Advisor:
Professor Emily M. Bender
Department of Linguistics

This thesis describes an analysis of valence-changing verbal morphology implemented as a library extending the LinGO Grammar Matrix customization system. This analysis is based on decomposition of these operations into rule components, which in turn are expressed as lexical rule supertypes that implement specific, isolatable constraints. I also show how common variations of these constraints can be abstracted and parameterized by their axes of variation. I then demonstrate how these supertypes can be recomposed in various combinations to provide broad coverage of the typological variation of valence change found in the world's languages. I evaluate the coverage of this library on several world languages that exhibit these phenomena.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, a most heartfelt thank you to my advisor, Emily M. Bender, not only for bringing me into the DELPH-IN fold, but also for her knowledge, advice, insight, and unflagging support and encouragement. Her commitment to nurturing the whole researcher in habits of mind and spirit is the very model of what an advisor should be. Thanks also to my committee reader, Sanghoun Song, for his infectious enthusiasm and thoughtful feedback. Any errors remaining in this thesis are entirely mine.

I would also like to thank the participants in DELPH-IN for their warm welcome, helpful feedback, and inspiration. I am privileged to count them as colleagues and friends. I would especially like to thank the standing committee members from other institutions: Dan Flickinger, Francis Bond, Stephan Oepen, Ann Copestake, John Carroll, and Hans Uszkoreit; their stewardship has made DELPH-IN the community it is.

Closer to home, I am grateful to the many friends without whose distractions this thesis might possibly have been completed sooner, but without whose support it would certainly never have been completed at all. Special thanks to my dear friend Josh Green, for his unfailing encouragement as well as his care and feeding of my sanity.

I also would like to thank my parents, Dan and Lynne, for instilling in me a love of learning and of language, and for a house filled with books as well as love.

Finally, I owe the greatest debt of thanks to my own family, canine and human. To Madeline, for her unconditional love and constant companionship. Most of all, to my partner, Micah, for his constant support every step of the way, through the (many) ups and downs, countless ways both small and large; thank you for always believing in me.

Chapter 1
INTRODUCTION

In this work I present an analysis of valence-changing verbal morphology in order to test two primary hypotheses: first, that a typologically-informed set of implemented valence-changing operations can cover a meaningful proportion of the incidence of valence change in the world's languages; and, second, that these valence-changing operations can be implemented in a "building-block" fashion by building up complete valence change operations from isolated, common elements that can be reused and recombined in varying combinations.

To evaluate these hypotheses I developed an implementation of valence-changing verbal morphology in the LinGO Grammar Matrix (Bender, Flickinger, & Oepen, 2002) and its customization system (Bender, Drellishak, Fokkens, Poulson, & Saleem, 2010). This implementation is based on decomposition of valence-changing operations into rule components, which in turn are expressed as lexical rule supertypes that implement specific, isolatable constraints. I also show how common variations of these constraints can be abstracted and parameterized by their axes of variation. I then show how these supertypes can be recomposed in various combinations to provide broad coverage of the typological variation of valence change found in the world's languages. I describe an implementation of this analysis as a library in the LinGO Grammar Matrix customization system, and evaluate its coverage on several world languages that exhibit these phenomena.

In Chapter 2 I begin by reviewing the LinGO Grammar Matrix and its customization system, and elaborate on the process of developing libraries to implement specific phenomena. I also briefly highlight the HPSG syntactic approach and its specific implementation in the Grammar Matrix, the relationship between valence and argument structure, and the MRS semantic

representation used in the Matrix. I then review the typological literature on valence change in Chapter 3, including the formation of broad categories of related valence-changing phenomena to inform my later analysis and implementation.

Chapter 4 consists of my analyses of valence-changing phenomena in HPSG as expressed in the Grammar Matrix. I also describe my decomposition of these phenomena into rule components and variable implementations, and the ways in which those components can be reassembled to implement specific valence-changing rules. These analyses are then revisited in Chapter 5, where I discuss how my theoretical analyses are translated into code that generates the appropriate Type Description Language (TDL) elements to implement the specified valence-changing operations.

In Chapter 6 I describe my process of evaluating my library against its design goals, beginning first with the development of pseudolanguage tests and their integration into the regression testing framework. I then show how I used several natural languages to illustrate valence change and identify errors and gaps in my implementation. Finally, I discuss my evaluation of my library against several held-out test languages which were not used in development and present the performance of my library in terms of its ability to support the variation of valence change found in the world's languages. In Chapter 7, I summarize this thesis along with possible directions for future work.

Chapter 2
BACKGROUND

In this chapter, I begin by providing an overview of the LinGO Grammar Matrix (Bender et al., 2002) and the Grammar Matrix customization system (Bender et al., 2010) on which this work is based. I also describe the general process of library development for the Grammar Matrix customization system, followed by a summary of the mapping from this template to the specific goals of this thesis. I then provide a brief overview of Head-driven Phrase Structure Grammar (HPSG; Pollard and Sag 1994), the theoretical grammatical basis underlying the Grammar Matrix, and in particular its approach to morphotactics, argument structure, and valence. Finally, I give a summary view of Minimal Recursion Semantics (MRS; Copestake, Flickinger, Pollard, and Sag 2005) and the DELPH-IN[1] joint reference formalism, Type Description Language (TDL; Copestake 2002a), used to implement the Grammar Matrix and Matrix-derived grammars.

2.1 The Grammar Matrix

The LinGO Grammar Matrix (Bender et al., 2002) is a resource that enables linguists to create implemented precision grammars. The core of the Grammar Matrix is a collection of types and constraints expected to be cross-linguistically useful, such as lexical and phrase rule types, feature geometry, and types implementing compositionality and long-distance dependency resolution. These analyses embed linguistic knowledge developed and tested by linguists and grammar writers over many years, in implementations of grammars at both large and small scales, in a framework that provides infrastructure and context for reuse in development of new grammars.

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

Beyond reuse and rapid development of new grammars, aspects of the *engineering* purpose of the Grammar Matrix, the Matrix also serves two scientific goals, as articulated by Bender et al. (2010): first, to support linguistic hypothesis testing through grammar engineering; and, second, to combine both breadth of typological research and depth of syntactic analysis into a single computational resource. The ability to rapidly create a basic grammar enables linguists to explore hypotheses about interesting linguistic phenomena and their interactions, and the depth and breadth of the completed analyses contained in the Matrix provide a sound linguistically-grounded starting point for these explorations. Examples of grammar engineering for linguistic hypothesis testing include assessing the completeness of an analysis of complex agreement patterns in Sahaptin [yak] (Drellishak, 2009) and evaluating competing analyses of auxiliaries in Wambaya [wmb] (Bender, 2010); further background and examples of hypothesis testing can be found in Bender and Langendoen 2010 and Bender 2008.

Grammar Matrix customization system

The Grammar Matrix customization system (Bender et al., 2010) combines a structured means of eliciting typological characteristics, validating responses for consistency, and using those choices to combine Matrix core grammar elements with stored analyses of various linguistic phenomena into a customized grammar. These stored analyses can include both static representations of cross-linguistically common phenomena as well as dynamically-generated implementations that embody language-specific variations. Elicitation is accomplished via a dynamic, iteratively-generated HTML questionnaire, which records the responses (while validating the consistency of both individual responses and their combination) in a structured choices file. This choices file is then processed by the customization script to produce the customized grammar. The system components and their relationships are shown in Figure 2.1 from Bender et al., 2010, p. 31.

The stored analyses of linguistic phenomena in the customization system are organized into conceptual "libraries". These libraries also provide elements of the questionnaire, customization routines, and validation logic associated with the phenomena analyses they control. Rep-

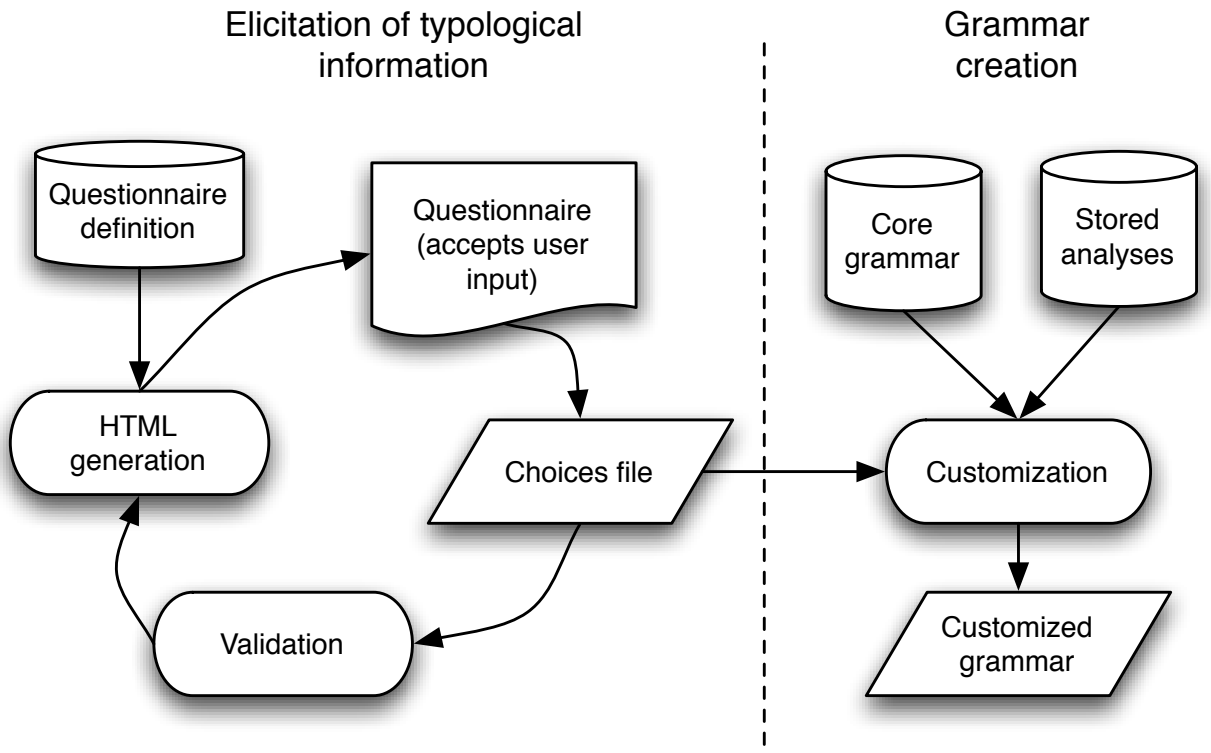


Figure 2.1: Customization system overview

resentative libraries include word order (Fokkens, 2010), sentential negation (Crowgey, 2012), argument optionality (Saleem, 2010; Saleem & Bender, 2010), and information structure (Song, 2014), among others. Libraries may also interact and depend on facilities provided by other libraries; libraries such as the one presented here that implicate morphology may have relatively tighter coupling to the morphotactics library, for example.

Library development process

The process of developing a library of stored analyses begins with identifying a phenomenon or set of related phenomena of interest and reviewing the typological literature describing it. From this review, the outlines of a library can be established, defining the range of elements that the library will and will not attempt to address. Once this scope is defined, the library writer can begin developing analyses of the in-scope phenomena and assessing potential implementation strategies.

An important part of this process is identifying the elements of the analyses that are determined by input from a grammar writer. These articulation points, and the range of available options for each, determine the information that needs to be conveyed in a choices file to the customization system. In many instances, the existing mechanisms provided by the customization system can be leveraged; for example, a new feature can be applied by the customization system's feature-implementing code. It may also be the case that the chosen approach does not readily fit into an existing code path and so the library developer may need to modify or add elements of the underlying customization system framework.

Once an initial design for choices file inputs is complete, including an understanding of any changes necessary to the customization system itself, the library developer can begin developing the questionnaire sections appropriate for the library. Many libraries will most naturally be suited to be presented via a new, dedicated section (page) of the questionnaire, while other libraries more logically add or modify the options available in other existing sections. The questionnaire and choices file structure are naturally interdependent and so as development progresses modifications to the choices structure may result in evolution of the questionnaire; by

the same token, as the questionnaire changes to better elicit the grammar writer's intent, the choices file structure will likely be adjusted as well.

The bulk of the development work itself remains in writing the code that, given the selected choices, adds the appropriate TDL fragments to the customized grammar. In addition to translating the theoretical analyses into TDL for implementation, the library writer also must address how the new functionality and TDL elements interact with existing parts of the system, including other libraries, the customization system, and even core type definitions provided by the Grammar Matrix. A key tool for the library developer in implementing this code, and especially being confident in the effect of modifications and interactions with other elements, is the regression testing framework.

Regression testing

The Grammar Matrix customization system includes a framework for managing and running suites of tests designed to exercise aspects of the system core and libraries (Bender, Poulson, Drellishak, & Evans, 2007). These test suites, usually created as part of library development, are comprised of a choices file, a set of test sentences marked with grammaticality judgments, and a set of gold-standard MRS results. For each selected test suite (or subset of test suites), the regression test framework first customizes a grammar from the choices file, parses the test sentences using the customized grammar, and compares the output MRS to the saved gold MRS, reporting any differences found. In this manner each regression test suite performs a complete, end-to-end test of the system's ability to create valid grammars that work as expected.

Most of the test suites currently included in the customization system implement so-called pseudolanguages, which are small abstract languages created by a developer to test a specific configuration of a phenomenon. For example, a developer might create pseudolanguages for both head-initial and head-final forms of the phenomenon being implemented, when that variation affects the grammar elements output by the library. In addition to pseudolanguages, test suites modeled on natural languages are also included, often drawn from illustrative and test languages used by library developers.

Taken in aggregate, these test suites represent a sort of minimal shape of the behavior of the customization system, across nearly all aspects of grammar customization. This broad perspective becomes extremely useful during development, when running all the regression tests[2] provides a developer with some degree of assurance that changes and additions to one area of the customization system have not caused unintentional errors in other areas.

Questionnaire and validation

One important aspect of the development of the questionnaire is the addition of validation checks that inform the grammar writer of invalid selections that block creation of a valid grammar. These validation tests can also provide warnings to the grammar writer where particular choices or combinations may have unexpected side effects, or where special considerations may apply for a given choice. The primary function of the validation code is to ensure that the customized grammar produced by the system is valid and self-consistent; as a rough functional standard, the grammar should be loadable into a DELPH-IN parser, such as the LKB (Linguistic Knowledge Builder; Copestake 2002b).

Goals and process

The purpose of this work is to apply this process to test two primary hypotheses: first, that a typologically-informed set of implemented valence-changing operations can cover a meaningful proportion of the incidence of valence change in the world's languages; and, second, that these valence-changing operations can be implemented in a "building-block" fashion by decomposing the complete valence change function into isolated, common elements that can then be recombined in varying combinations to produce the final operations.

To test these hypotheses, I constrained my scope to verbal morphology and reviewed the typological literature on valence change. I then identified broad categories of valence-changing operations and developed HPSG analyses of the phenomena, breaking them down in the process

[2] At this writing, there are approximately 450 distinct regression test suites.

into isolated elements that could be recombined in various ways to provide broad coverage of the space of typological variation. I then implemented these analyses in the Grammar Matrix customization system by developing questionnaire questions to elicit appropriate configuration parameters, establishing the structure of the relevant choices file elements, and writing the code to validate the combined choices and produce the appropriate output.

In developing the library, I created pseudolanguage test suites, designed to exercise the specific operations I developed, and added them to the set of regression tests. I also leveraged three illustrative languages to confirm my design against the valence change found in real-world languages, making changes to the library as necessary to correct errors and fill in gaps in the implementation. Finally, I attempted to model (with my library) the valence-changing morphology in five held-out languages in order to assess the degree to which my library, as implemented, was successful and achieved a meaningful breadth of coverage.

In this section, I have summarized the library development process and the components of the Grammar Matrix and customization system. In the next section, I briefly describe the theoretical basis of these elements and summarize the aspects of the Grammar Matrix relevant to the implementation of valence and valence change.

2.2 Head-driven phrase structure grammar

The Grammar Matrix is rooted in the theoretical framework of Head-driven Phrase Structure Grammar (HPSG; Pollard and Sag 1994). The foundation of HPSG is the representation of each linguistic sign as a typed feature structure. This typed feature structure (Carpenter, 1992) is a structured object consisting of defined attributes, or features, the values of which are themselves other typed feature structures. The type of a feature structure determines which features appear in it. Thus, the entire feature structure of a sign forms a directed graph, where each node is a feature structure, and each edge is labeled with a feature name. Each node in the graph can be reached by following a sequence of labeled edges from the root. However, this path need not be unique; two (or more) paths through different feature structures may reach the same node.

Disjoint feature structures can be combined, or *unified*, where their graphs (or a graph and a subgraph) are isomorphic. Unification is constrained such that the types of each node must be compatible; that is, that (a) the features present at the node must be valid for both node types, and (b) the types of each value the nodes have in common are also compatible. In this way the constraints expressed by each feature structure are satisfied. Note that this definition is recursive: each node must unify as well as each each descendant node.

A grammar in this paradigm is comprised of the following main elements:

- **lexical types**, constraints inherited by words in the lexicon
- **lexical rule types**, constraints on how stems give rise to inflected and derived forms
- **grammar rule types**, constraints on how words combine into phrases, and how phrases combine
- **foundational types**, types that constrain feature values (*e.g.*valid values of the CASE feature)
- **instances**, instantiations of lexical types and lexical and grammar rule types

This brief description illustrates two distinctive attributes of HPSG,[3] as described in Sag, Wasow, and Bender 2003, Chapter 9. First, grammars are based on constraint satisfaction through unification (as contrasted with a transformational approach). Second, the grammar's view of syntax is strongly lexical: constraints originate with instances of lexical types and a distinction is made between word-internal rules and syntactic rules, the latter having no access to the former.

The Grammar Matrix, and other tools compatible with DELPH-IN grammars, implement a restricted formalism, described in Copestake (2002a) and referred to as the DELPH-IN joint reference formalism, that significantly limits the available operations on feature structures. For example, the Joint Reference Formalism disallows set-valued features and relational constraints, and all structures must be acyclic. One implication of the Joint Reference Formalism with particular relevance to this work is that the prohibition on relational constraints excludes a list-append operation. As I show in more detail in later chapters, this affects the implementation of operations involving and removing arguments on valence lists.

---

[3]These attributes also apply to other grammar approaches in the same tradition.

Valence and argument structure

In the revised[4] conception of HPSG, the valence of a particular sign—that is, the specification of what other signs it must combine with to become saturated— is conveyed by the SUBCAT feature, which is defined as the append of the SUBJ, SPR, and COMPS lists (Pollard & Sag, 1994, p. 375). Subsequently, Manning and Sag (1998) proposed a modification whereby SUBCAT became a means to express the argument structure of a lexical sign, and specifically as a distinct entity from the SUBJ, SPR, and COMPSvalence lists. The SUBCAT feature was renamed to ARG-ST to indicate this revised role.

This separation and its concomitant materialization of the mechanisms for linking argument structure and valence lists, making them available for manipulation, is essential to the implementation of valence change in this work. As I describe in more detail in Chapter 3, operations such as the passive rely on changing the relationship between syntactic and semantic roles played by a verb's arguments.

Morphotactics

The Grammar Matrix customization system includes mechanisms for implementing morphosyntax, including the obligatoriness, ordering, and co-occurrence of position classes, and the definition and instantiation of lexical rules to implement inflectional (and, to a limited degree, derivational) morphology. The original morphotactics library was developed by O'Hara (2008), with argument optionality added by Saleem (2010). The current morphotactics framework is the result of significant modification and improvement by Goodman (2013).

Minimal Recursion Semantics

The preceding sections have given an overview of the syntactic formalism underlying the Grammar Matrix and Matrix-derived grammars. In this section I turn briefly to the semantics pro-

[4]With respect to earlier chapters of Pollard and Sag 1994.

duced[5] by these grammars. The Grammar Matrix, along with other DELPH-IN compatible grammars, uses Minimal Recursion Semantics (MRS; Copestake et al. 2005) as the semantic representation. MRS can be naturally expressed in terms of feature structures and so is integrated into the HPSG mechanisms and feature structures of the Grammar Matrix.

In MRS, the primary unit of interest for semantics is the elementary predication (EP), which is a single relation and its arguments, identified by a label. Typical lexical items contribute a single EP (though some have more and some have none); phrase structure rules construct a bag of EPs associated with the phrase by appending the bags of EPs of all the phrase daughters, and may contribute EPs themselves. EPs are never embedded in other EPs, but are instead grouped as flat elements in a bag (as opposed to a set; EPs may be repeated). This flat representation is underspecified as to scope, so an additional set of constraints are applied that define a restricted set of ways in which EPs can be related via scope relations. Scopal arguments are expressed via handle relationships, in which a handle is equal, modulo quantifiers, to a label. This relationship, denoted as *qeq* or $=_q$, allows semantic composition to be defined simply while preserving the scope constraints that could contain intervening quantifiers. This distinction between scopal and non-scopal relationships is essential to expressing certain valence-changing phenomena such as the causative, which I analyze and discuss in more detail in my analysis (Chapter 4).

2.2.1 Summary

In this chapter, I described the Grammar Matrix and the Grammar Matrix customization system, along with an overview of the library development process and its relationship to the goals of this thesis. I also presented a brief summary of the theoretical underpinnings of the Grammar Matrix and customization system libraries, particularly those elements relevant to valence change. In the next chapter, I review the typological literature on valence change, and begin to establish broad categories of valence-changing phenomena for analysis and implementation.

---

[5] Or consumed; a hallmark of Matrix-derived grammars is the ability to generate syntactically-valid sentences from semantic representations.

Chapter 3

REVIEW OF LITERATURE

In this chapter, I review some of the relevant literature on valence and valence change, as well as the aspects of the Grammar Matrix implicated in their implementation.

Valence, by analogy to the valence of atoms in chemistry (Tesnière, 1959), refers to the number of core syntactic arguments a verb in a given clause type takes. All human languages have both intransitive and transitive clauses: intransitive clauses have a single argument, the subject (denoted S); transitive clauses have two arguments, the transitive subject (denoted A) and the transitive object (denoted O) (Dixon, 1979). *Subject* as used here refers to the first, or least-oblique, core argument to a verb, usually associated with the semantic role of agent, while *object* refers to a complement of a verbal head, often associated with the semantic role of patient or theme. Many languages also allow an additional argument in the core of a transitive clause, such as the recipient or beneficiary of an action (e.g. the dative alternation in English [eng]), denoted as E ('extension to core'), and a few languages also allow an extended intransitive clause type (Dixon & Aikhenvald, 2000, p. 3). To summarize these argument patterns:

intransitive	S		
extended intransitive	S		E
transitive	A	O	
extended transitive	A	O	E

Note that in this scheme transitivity is distinct from valence: both plain transitive and extended intransitive are bivalent. This distinction, however, is not relevant for the analysis or implementation of this thesis and so E arguments are not addressed.

3.1 Typological survey of valence change

Many languages permit verbal derivations that alter the argument structure of verbs, either increasing or decreasing the valence and changing the relationship of realized arguments to syntactic roles. I describe the cross-linguistic range of these operations below; following the broad conceptual framework provided by Haspelmath and Müller-Bardey (2004) (henceforth H&MB) I group the operations first by whether they reduce or increase valence and second by whether they affect the subject or object. I also retain their focus on verbal valence-changing morphology, excluding e.g., periphrastic constructions.

3.1.1 Valence-reducing operations

Subject-removing

The primary types of subject-removing operation to consider are the anticausative and the passive. Both remove the subject (A) and move the former object (O) into the subject position; the essential distinction between them is that the anticausative removes the A argument entirely, while the passive merely moves it to the periphery (H&MB). The Turkish [tur] anticausative and passive in Mam [mam] (Mayan family), are illustrated in (1) and (2), respectively:

(1) a. Anne-m kapı-yı aç-tı
mother-1SG door-ACC open-PAST(3SG)
'My mother opened the door.' [tur]

b. Kapı aç-tı-dı
door open-ANTIC-PAST(3SG)
'The door opened.' [tur] (H&MB, p. 5)

(2) a. ma ch-ok t-b'iyo-'n Cheep kab' xjaa
PAST 3PL+O-DIRectional 3SG+A-hit-DIR José two person
'José hit two people.' [mam]

b. ma chi b'iy-eet kab' xjaa (t-u'n Cheep)
PAST 3PL+S hit-PASS two person 3SG-REL/AGENT José
'Two people were hit (by José).' [mam]
(England, 1983, in Dixon & Aikhenvald, 1997, p. 75)

A related subject-removing operation is the resultative (Comrie & Nedjalkov, 1988), which both removes the subject and changes the verb to refer to a resulting state rather than the causing event. The contrast is illustrated between the anticausative (3b) and resultative in Russian [rus] (3c) below:

(3) a. Mira zagyvaet dver'
Mira closes door
'Mira is closing the door.' [rus]

b. Dver' zakryvaet-sja
door closes-REFL
'The door is closing.' [rus]

c. Dver' zakry-ta
door close-PRTCP
'The door is closed.' [rus] (H&MB, pp. 5–6)

Object-removing

Analogous to the anticausative, the object-removing operation where the object O is completely removed is referred to as the deobjective (H&MB) or the absolutive antipassive (Dayley, 1989, as cited in H&MB), and is illustrated by Ainu [ain] (4) below. A related form, the "potential deobjective," expresses disposition of an agent rather than a real action; however, this semantic distinction is not relevant to this analysis.

(4) a. Sake a-ku
sake 1SG.TR-drink
'I drink sake.' [ain]

b. I-ku-an
DEOBJ-drink-1SG.INTR
'I drink.' [ain] (Shibatani, 1990, in H&MB, p. 3)

The deaccusative (H&MB) or antipassive[1] (Dixon & Aikhenvald, 2000) is similar, but instead of completely removing the underlying O argument, moves it out of the core to the periphery, as illustrated by the Hungarian [hun] deaccusative in (5).

(5) a. Az orvos szán-ja a beteg-et
the doctor pity-3SG the patient-ACC
'The doctor pities the patient.' [hun]

b. Az orvos szán-akoz-ik a beteg-en
the doctor pity-DEACC-3SG the patient-SUPERESS
'The doctor feels pity for the patient.' [hun] (Károly, 1982, in H&MB, p. 4)

Reflexive and reciprocal

Two special cases of valence reduction are the reflexive and reciprocal.[2] The reflexive expresses an action performed by the subject on the subject, as in Modern Greek [ell] (6), whereas the reciprocal expresses an action performed mutually by (necessarily plural) subjects on or to each other, illustrated via Turkish in (7). In both cases, the object is not overtly expressed as a syntactic argument, but is coindexed semantically with the subject.

(6) a. O Axmed ksíri-s-e ton Péro
ART Ahmed shave-AOR-3SG ART Pedro
'Ahmed shaved Pedro.'

b. O Pero ksirí-s-tik-e
ART Pedro shave-AOR-REFL-3SG
'Pedro shaved (himself).' [ell] (H&MB, p. 6)

---

[1] Dixon and Aikhenvald use 'antipassive' to refer to both constructions, noting simply that "the [underlying O] argument may be omitted."

[2] Cross-linguistically, valence-reducing derivation is only one strategy for expressing reflexive/reciprocal; the other common strategy, placing a reflexive or reciprocal pronoun in the O slot (universally O and not A; see Dixon & Aikhenvald, 2000, p .11), maintains the bivalent argument structure and so is not relevant to this thesis.

(7) a. Sara Farid-i sev-iyor
Sara Farid-ACC love-IMPF
'Sara loves Farid.'

b. Dost-lar sev-is-iyor-lar
friend-PL love-RECIP-IMPF-PL
'The friends love each other.' [tur] (H&MB, p. 7)

3.1.2 Valence-increasing operations

Subject-adding

Cross-linguistically the most common valence-changing category (Bybee, 1985), the causative adds a new subject (A), the causer of the event described by the verb. The addition of a causer to an intransitive verb can simply move the underlying subject (S) into an object (O) position, as illustrated by the Vengo [bav] (Grassfields Bantu) causative in (8):

(8) a. nw nìi táa nìì
he enter in house
'He entered the house.' [bav]

b. m nìi-s nw táa nìì
I enter-CAUS him in house
'I made him enter the house.' [bav] (Schaub, 1982, in H&MB, p. 11)

The situation with underlying transitive verbs is more complex, as there are different strategies for dealing with the underlying subject (causee), given the presence of an already-existing direct object (O). H&MB identify three such strategies, illustrated in (9): (9a) causee as an indirect object, as in Georgian [kat], (9b) causee as instrumental phrase, as in Kannada [kan], and (9c) causee as second direct object, as in Imbabura Kwicha [qvi].

(9) a. Mama-m Mzia-s daanteb-in-a cecxli
father-ERG Mzia-DAT light-CAUS-AOR:3SG fire(ABS)
'Father made Mzia light the fire.' [kat] (Harris, 1981, in H&MB, p. 12)

b. Raamanu manga-gal-inda Siite-yannu huduki-si-danu
Rama(NOM) monkey-PL-INSTR Sita-ACC search-CAUS-3SG
'Rama had the monkeys search for Sita.' [kan]
(Cole & Sridhar, 1977, in H&MB, p. 12)

c. Juzi-ka Juan-ta ruwana-ta awa-chi-rka
José Juan-ACC poncho-ACC weave-CAUS-3SG
'José made Juan weave a poncho.' [qvi] (Cole, 1982, in H&MB, p. 12)

Other subject-adding constructions are structurally similar to the causative, such as the affective ('indirect passive') in Japanese [jpn], shown in (10b):

(10) a. John-ga Mary-ni tokei-o nusum-ase-ta
John-NOM Mary-DAT watch-ACC steal-CAUS-PST
'John let Mary steal a watch.' [jpn]

b. John-ga Mary-ni tokei-o nusum-are-ta
John-NOM Mary-DAT watch-ACC steal-AFF-PST
'John was affected by Mary stealing (his) watch.' [jpn]
(Washio, 1995, in Wunderlich, 2015, p. 20)

A crucial aspect of the causative and similar constructions is the addition of a new EP which functions as a scopal operator with respect to the verb's own EP and takes as an argument the added participant. This is distinguished from the applicative (below), which is non-scopal and does not affect semantic roles.

Object-adding

Object-adding constructions can collectively be grouped under the term 'applicative,' which subsumes a broad variation in potential roles for the added structural argument. The prototypical applicative is the benefactive, as demonstrated in the Indonesian [ind] alternation in (11):

(11) a. Ali memi telefisi untuk ibu-nja
Ali TR.buy television for mother-his
'Ali bought a television for his mother.' [ind]

b. Ali mem-beli-kan ibu-nja telefisi
Ali TR-buy-APPL mother-his television
'Ali bought his mother a television.' [ind]
(Chung, 1976, in Wunderlich, 2015, p. 21)

In the Bantu languages, applicatives can serve many different functions; for example, the Kinyarwanda [kin] examples in (12) below illustrate benefactive, possessor-raising, and instrumental applicatives:

(12) a. umugóre y-a-som-e-ye umwáana igitabo
woman 3SG-PST-read-APPL-PERF child book
'The woman read the book to the child.' [kin]

b. umugabo a-ra-kikir-ir-a umugóre umwáana
man 3SG-PRES-hold-APPL-IMPF woman child
'The man is holding the woman's child.' [kin]

c. umugóre y-Ø-uhag-iish-ije umwáana isábune
woman 3SG-PST-wash-APPL-PERF child soap
'The woman washed the child with soap.' [kin]

d. umugóre y-Ø-uhag-iish-ije umkukoóbwa umwáana
woman 3SG-PST-wash-CAUS-PERF girl child
'The woman made the girl wash the child.' [kin]

(Polinsky & Kozinsky, 1992, in Wunderlich, 2015, p. 24)

An unusual case is Halkomelem [hur] (Salish family), where so-called 'psych applicatives' (Gerdts & Kiyosawa, 2005) can attach to an intransitive psychological predicate and form a transitive clause, with the added object argument serving as the stimulus. This relational suffix (REL) is illustrated below in (13a) and contrasted with the (intransitive) oblique NP form in (13b) and the causative[3] in (13c):

(13) a. niʔ cən siʔsiʔ-meʔ-t kʷθə sqʷəmey
AUX 1SUB frighten-REL-tr DET dog
'I was frightened at the dog.' [hur]

b. niʔ cən siʔsiʔ ʔə kʷθə snəxʷəɬ
AUX 1SUB frighten OBL DET canoe
'I was frightened at the car.' [hur] (Gerdts & Kiyosawa, 2005, p. 339)

c. niʔ cən siʔsiʔ-stəxʷ kʷθə sməyəθ
AUX 1SG.SUBJ frighten-CAUS:3OBJ DET deer
'I frightened the deer.' [hur] (Gerdts & Kiyosawa, 2005, p. 334)

[3] In the causative the stimulus is the subject, which is not true in the psych applicatives.

The relative applicative can also be applied to transitive psychological predicates, with the effect of maintaining the existing syntactic relations but changing the semantic roles: the subject agent becomes the experiencer and the object experiencer becomes the stimulus. This is illustrated below in (14):

(14) a. c'q'-ət č ce k^{w}θə nəc'əwməx^{w} i ce tecəl
surprise-TR 2SUBJ FUT DET AUX FUT arrive
'You will surprise the visitors when they arrive.' [hur]

b. c'əq'-me-t č ce k^{w}θə nəc'əwməx^{w} i ce tecəl
surprise-REL-TR 2SUBJ FUT DET AUX FUT arrive
'You will be surprised at the visitors when they arrive.' [hur]
(Gerdts & Kiyosawa, 2005, p. 334)

In this section I have illustrated the broad scope of variation in valence change cross-linguistically and have shown how different forms of valence change can be grouped together by their effect on valence. These groupings highlight a great deal of syntactic similarity, despite widely varying semantic implications. I address these commonalities in depth in Chapter 4, where they form the basis for my analysis.

3.1.3 Typology of valence-changing morphs

In this discussion, I have focused on valence-changing operations that correspond to some morphological marking of the construction. Indeed, Dixon and Aikhenvald (2000) include this formal marking criterion in each of their typological definitions. This excludes from consideration, for example, the well-known English 'dative shift' in (15) (cf. Indonesian in (11)):

(15) a. Sarah gave a mango to Farid
Sarah give\PST a mango to Farid
'Sarah gave a mango to Farid.' [eng]

b. Sarah gave Farid a mango
Sarah give\PST Farid a mango
'Sara gave Farid a mango.' [eng] (after H&MB, p. 2)

The actual morphological expression of valence change can vary widely cross-linguistically, including prefixes, suffixes, infixes, circumfixes, and even reduplication (attested e.g. in Cavineña [cav]; see Guillaume 2014).

For the purposes of this work, however, the surface morphophonological form (including zero-expressed morphemes) is assumed to have been transformed into a straightforward mapping onto morphosyntactic position classes, potentially by means of an external morphophonological analyzer.[4]

3.1.4 Summary

In this section I have reviewed the range of valence change operations in scope for this thesis. In the following chapter, I will present the analyses of these operations, grounded in the Grammar Matrix as described in the next section.

[4]See Bender and Good 2005 for the theoretical and practical basis for this approach.

Chapter 4

ANALYSIS

In this chapter I develop analyses of valence-changing operations and show how these analyses can be decomposed into rule components, expressed as lexical rule supertypes implementing specific, isolatable constraints. I also demonstrate how these rule components can then be re-composed via type inheritance to form lexical rule types that, when instantiated in a grammar, implement the desired valence-changing operation. Throughout this chapter, I describe the analyses I developed during the course of the initial development. Some of these analyses were later refined through evaluation with illustrative languages as described in Section 6.2; I have attempted to note in this chapter those analyses where where significant changes were made.

4.1 Approach

The overall approach I followed was to decompose the high-level, linguistically-significant valence-changing operations into their component operations on feature structures. These individual component operations can then be selected by the customization system and composed to achieve the high-level result. The components I selected to analyze and implement included addition and removal of subjects and objects, case constraints and alternations, and argument reordering. In the following sections, I discuss the HPSG analysis of each of the component operations and the resulting effects on the feature structures. I discuss my analyses of valence-reducing phenomena, beginning with subject-removing and subject-demoting operations, and continuing on in turn to object-removing operations. Next I address valence-increasing phenomena, first analyzing object-adding operations and their decomposition into building blocks, and then turning finally to subject-adding operations and their corresponding decomposition.

Throughout this chapter and in those that follow, I depart slightly from the terminology in H&MB and refer to these operations in terms of their *syntactic* effect (e.g., subject-removing rather than agent-removing). This serves two purposes: one, it foregrounds the structural nature of the operations, and, second, better reflects the semantically-neutral feature names (ARG1, ARG2, etc.) normally used in MRS as deployed in DELPH-IN feature structures (see e.g. Copestake et al., 2005, p. 305).

Beginning with valence-reducing operations in the next section, and continuing with valence-increasing and valence-modifying operations in subsequent sections, I present examples of the phenomena along with HPSG analyses of their implementation in lexical rule types. As the complexity of the operations increases, I further discuss the constraints that make up the rule types and how those individual constraints may be extracted into a type hierarchy for reuse.

4.2 Valence reduction

4.2.1 Subject removal

H&MB distinguish a *true* subject-removing operation, such as the anticausative, from the passive, whereby the erstwhile subject is demoted to an oblique position, and I maintain this distinction in this analysis. (I analyze the passive separately as a subject-demoting operation in section 4.4 below.) A true subject-removing operation, the anticausative, is illustrated in the Turkish [tur] example (1) given in Chapter 3 (repeated here as (16)).

(16) a. Anne-m kapı-yı aç-tı
mother-1SG door-ACC open-PAST(3SG)
'My mother opened the door.' [tur]

b. Kapı aç-tı-dı
door open-ANTIC-PAST(3SG)
'The door opened.' [tur] (H&MB, p. 5)

This operation can be analyzed concisely as a lexical rule that attaches to verbs (or the output of other appropriate lexical rules) and leaves the rule daughter's SUBJ as *unexpressed*.[1] In addition, the first item[2] on the daughter's COMPS list is coindexed with the mother's SUBJ, with the remaining COMPS elements forming the mother's COMPS list. Finally, to allow the output of this lexical rule to be properly handled by external control or raising predicates, the XARG value of the output's HOOK is coindexed with the new subject. This lexical rule is illustrated in the AVM shown in (17).

(17)
$$\left[\begin{array}{ll} \textit{subj-rem-op-tr-lex-rule} & \\ \textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val} & \left[\begin{array}{ll} \textsc{subj} & \langle \left[\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index}\quad \boxed{1}\right] \rangle \\ \textsc{comps} & \boxed{2} \end{array}\right] \\ \textsc{c-cont}\,|\,\textsc{hook}\,|\,\textsc{xarg} & \boxed{1} \\ \textsc{dtr}\,|\,\textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val} & \left[\begin{array}{ll} \textsc{subj} & \langle\, \textit{unexpressed} \,\rangle \\ \textsc{comps} & \left[\begin{array}{ll} \textsc{first} & \left[\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index}\quad \boxed{1}\right] \\ \textsc{rest} & \boxed{2} \end{array}\right] \end{array}\right] \end{array}\right]$$

In the case of an intransitive verb, subject removal occurs in constructions such as the impersonal passive, as illustrated in (18).[3] The corresponding lexical rule, illustrated in (19), is simpler than for the transitive and merely saturates the SUBJ valence slot, leaving the daughter's SUBJ, again, as *unexpressed*. Although in this discussion I have treated these as distinct rules,

---

[1]The Grammar Matrix type *unexpressed* is used here, following the Matrix implementation of the threading analysis of Bouma, Malouf, and Sag (2001), to ensure that the missing subject does not break difference list operations on non-local features.

[2]Although conceptually it ought to be possible to promote either the first or last element into subject position, the presumption that the first element is always the one promoted is sufficiently strong that Pollard and Sag (1994, p. 119) take it as axiomatic across a number of theoretical frameworks.

[3]Some languages require an expletive subject, as in German [deu]:

(i) Es wird getanzt
it is.PASS dance.PTCP
'There is dancing.' [deu]

In the present work I only address the cases where the subject is fully removed.

they can also be viewed as variations on the intransitive-transitive axis of a single abstract rule. (I expand on this concept further in section 4.3.1 below.)

(18) Dün sorun üzerinde düşün-ül-dü.
yesterday problem about think-PASS-PAST.3PER
'Yesterday it was thought about the problem.' [tur]
(Nakiboglu-Demiralp, 2001, p. 131)

(19)
$$\begin{bmatrix} \textit{subj-rem-op-itr-lex-rule} & \\ \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{subj} & \langle\ \rangle \\ \textsc{dtr} \mid \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{subj} & \langle\ \textit{unexpressed}\ \rangle \end{bmatrix}$$

4.2.2 Object (complement) removal

The deobjective can be analyzed similarly, operating instead on the COMPS valence list. Object-removing has the additional complication, however, of needing to remove only a single item from a list that may have more than one complement, such as in the case of ditransitives.[4] Accordingly, the REST of the COMPS list is copied to the rule output as the new COMPS list.[5] Object-removing also requires, naturally, that the daughter be at least transitive, as is evidenced by the daughter's COMPS structure.

This rule is illustrated in (20) below:

(20)
$$\begin{bmatrix} \textit{obj-rem-op-lex-rule} & & \\ \textsc{synsem} & \left[\textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} \quad \boxed{1}\right] & \\ \textsc{dtr} & \left[\textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} \quad \begin{bmatrix} \textsc{first} & \textit{unexpressed} \\ \textsc{rest} & \boxed{1} \end{bmatrix}\right] \end{bmatrix}$$

[4] The customization system does not currently support creating ditransitives directly.

[5] Similarly to the subject-removing operation, I extended here the presumption that only the first element can be a target. However, in modeling Zulu as an illustrative language (see section 6.2.3) I found that this rule needed to be parameterized to target either the first or second complement.

4.3 Valence increase

4.3.1 Object addition

Object addition covers the general category of the applicative, which subsumes a variety of different types of oblique roles cross-linguistically, including the instrumental and benefactive (H&MB, p. 7). In adding an argument, there are several underlying operations in my analysis:

- Adding an argument to the COMPS list. Note that, cross-linguistically, the added argument can added either more- or less-obliquely to the verb's existing dependencies (i.e., at the head or tail of the COMPS list)
- Constraining the added argument (or promoted subject), e.g. to be an NP or PP (HEAD *noun* or *adp*), or applying a CASE constraint
- Appending the new argument's non-local dependencies to the rule mother's list
- Contributing an added elementary predication (EP) via C-CONT
- Linking the new EP's ARG1 to the daughter's INDEX
- Linking the new EP's ARG2 to the new argument's INDEX

The first two of these operations are directly grounded in the addition of a new argument and are straightforward; appending the new argument's non-local dependencies simply maintains the threading analysis of Bouma et al. (2001) and is similarly straightforward. This operation is discussed in more detail along with the example in (27) below.

The addition of a new EP to the rule output is not as straightforward and requires some additional discussion. To motivate this analysis, consider the example of the benefactive from Indonesian in (21). In this example, the addition of the benefactive applicative suffix *-kan* in (21b) adds an argument position to the verb, which is filled by *perempuan itu* "the woman". Note that the added argument is in fact an object, as it permits passivization as in (21c).

(21) a. Orang itu masak ikan untuk perempuan itu
man DEF cook fish for woman DEF
'The man cooked fish for the woman.' [ind]

b. Orang itu memasakan perempuan itu ikan
Orang itu me-masak-kan perempuan itu ikan
man DEF TR-cook-BEN woman DEF fish
'The man cooked the woman fish.' [ind]

c. Perempuan itu dimasakan ikan oleh orang itu
Perempuan itu di-masak-kan ikan oleh orang itu
woman DEF PASS-cook-BEN fish by man DEF
'The woman was cooked fish by the man.' [ind] (Chung, 1976, p. 58)

Notionally, the benefactive is adding a third semantic argument to the verb, which would add a hypothetical ARG3 to the EP contributed by the verb;[6] however, this would seem to violate the principles of semantic composition in Copestake et al. (2005); namely, that composition consists solely of *concatenation* of daughter RELS values, not modification. More concretely, there is no EDP-modifying operation available within the algebra of Copestake, Lascarides, and Flickinger (2001).[7]

The solution is to have the lexical rule contribute a new EP, which takes both the EP contributed by the verb and the additional syntactic argument as semantic arguments. The predicate value for this new EP will provide the particular species of applicative (e.g., benefactive, as here). This new EP contributes its own event and takes as its arguments the respective indexes of the input and the added argument. In this analysis I treat the added arguments as non-scopal, with no intervening handle relationships; this contrasts with my analysis of subject addition (see section 4.3.2 below).

The introduction of a new event by this EP differs from the analysis of the benefactive presented by Müller (2017);[8] by maintaining a separate event variable for the added EP, my analysis here makes the relation contributed by the EP potentially available for modification separately

[6] Or a lexical rule which has previously been applied to the input.

[7] Although this principle is generally strongly-embraced by DELPH-IN grammars, it is not entirely settled whether this necessarily should be as strictly applied within lexical rules (see e.g. Copestake, Lascarides, and Bender 2016 and Bender 2015); it also is not prohibited by the DELPH-IN joint reference formalism (Copestake, 2002a). My analyses in this work are not frustrated by adhering to this principle, so I retain it as applying throughout a grammar.

[8] Müller 2017 does not make an explicit argument for the shared-event approach.

from the event of the main verb (as in the English [eng] periphrastic form *Kim read the book, probably for Sandy*). The MRS resulting from this analysis is shown in (22):

(22)
$$\left[\text{RELS}\ \left\langle \begin{bmatrix}\textbf{\_memi\_v\_buy}\\ \text{ARG0}\ \boxed{4}\ \textit{event}\\ \text{ARG1}\ \boxed{1}\\ \text{ARG2}\ \boxed{2}\end{bmatrix}, \begin{bmatrix}\textbf{named}\\ \text{ARG0}\ \boxed{1}\end{bmatrix}, \begin{bmatrix}\textbf{\_telefisi\_n\_TV}\\ \text{ARG0}\ \boxed{2}\end{bmatrix}, \begin{bmatrix}\textbf{\_ibu\_n\_mother}\\ \text{ARG0}\ \boxed{3}\end{bmatrix}, \begin{bmatrix}\textbf{benefactive}\\ \text{ARG0}\ \textit{event}\\ \text{ARG1}\ \boxed{4}\\ \text{ARG2}\ \boxed{3}\end{bmatrix} \right\rangle\right]$$

With all these elements combined, a complete rule implementing the benefactive (with the argument being added less-obliquely, in this example) is illustrated in (23).

(23)
$$\left[\textit{benefactive-lex-rule}\right.$$
$$\text{SYNSEM}\ \left[\begin{array}{l}\text{LOCAL}\,|\,\text{CAT}\,|\,\text{VAL}\,|\,\text{COMPS}\ \left\langle \boxed{1}, \left[\begin{array}{l}\text{LOCAL}\ \left[\text{CAT}\ \left[\begin{array}{ll}\text{HEAD} & \textit{noun}\\ \text{VAL} & \left[\begin{array}{ll}\text{SPR} & \langle\,\rangle\\ \text{COMPS} & \langle\,\rangle\end{array}\right]\end{array}\right]\right]\\ \text{CONT}\,|\,\text{HOOK}\,|\,\text{INDEX}\ \boxed{2}\\ \text{NON-LOCAL}\ \left[\begin{array}{ll}\text{SLASH} & \boxed{3}\\ \text{QUE} & \boxed{4}\\ \text{REL} & \boxed{5}\end{array}\right]\end{array}\right]\right\rangle\\ \text{NON-LOCAL}\ \left[\begin{array}{ll}\text{SLASH} & \boxed{7}\oplus\boxed{3}\\ \text{QUE} & \boxed{8}\oplus\boxed{4}\\ \text{REL} & \boxed{9}\oplus\boxed{5}\end{array}\right]\end{array}\right]$$
$$\text{C-CONT}\ \left[\begin{array}{l}\text{RELS}\ \left\langle !\ \left[\begin{array}{ll}\textit{event-relation}\\ \text{PRED} & \textit{“benefactive”}\\ \text{ARG1} & \boxed{6}\\ \text{ARG2} & \boxed{2}\end{array}\right]\ !\right\rangle\\ \text{HCONS}\ \langle !\ \ !\rangle\end{array}\right]$$
$$\text{DTR}\ \left[\begin{array}{l}\textit{verb-lex}\\ \text{SYNSEM}\ \left[\begin{array}{ll}\text{LOCAL} & \left[\begin{array}{ll}\text{CAT}\,|\,\text{VAL}\,|\,\text{COMPS} & \boxed{1}\\ \text{CONT}\,|\,\text{HOOK}\,|\,\text{INDEX} & \boxed{6}\end{array}\right]\\ \text{NON-LOCAL} & \left[\begin{array}{ll}\text{SLASH} & \boxed{7}\\ \text{QUE} & \boxed{8}\\ \text{REL} & \boxed{9}\end{array}\right]\end{array}\right]\end{array}\right]$$
$$\left.\right]$$

rule component	varies by
added argument	position (obliqueness), number of existing args
constraint on new argument	position (obliqueness), constraint (*e.g.* case, head)
non-local dependencies	position (obliqueness)
new EP's PRED value	predicate
new EP's ARG1	does not vary
new EP's ARG2	position (obliqueness)

Table 4.1: Rule component axes of variation (benefactive)

This rule, however, in combining the distinct operations identified above, obscures common elements that can be reused for other similar object-adding operations. Reviewing the five operations, it is evident that they vary along different axes, as summarized in Table 4.1.

This leads to a simplification and optimization: in the same way that the intransitive and transitive forms of subject removal (see section 4.2.1 above) can be viewed as variants of an abstract analysis along the transitivity axis, these building-block operations can also be treated as being parameterized along their their axes of variation and then combined to make the final rule type.

Concretely, taking these operations in turn, the first operation (adding the argument) needs to have variants for adding an argument: (a) to intransitive or transitive verbs, and (b) at the front or end of the COMPS list. That is, the lexical rule type implementing each of the component operations can be viewed as the output of a function:

$$f : tr \in \{intrans, trans\} \times pos \in \{front, end\} \rightarrow lrt$$

To illustrate this variation, the rule type at (24) adds an argument at the *end* of the COMPS list for an *intransitive* verb,[9] and the rule at (25) adds an argument at the *front* of the COMPS list for a *transitive* verb and links the INDEX of that argument to its second semantic argument (ARG2).

[9]Naturally, for an intransitive input there is no difference between the front and end of the COMPS list. The same rule would be generated for either specification; formally, $f(intrans, front) \equiv f(intrans, end)$.

(24)
$$\begin{bmatrix} \textit{added-arg2of2-lex-rule} & \\ \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} & \left\langle \begin{bmatrix} \textsc{local} & \begin{bmatrix} \textsc{cat} \mid \textsc{val} & \begin{bmatrix} \textsc{spr} & \langle\,\rangle \\ \textsc{comps} & \langle\,\rangle \end{bmatrix} \\ \textsc{cont} \mid \textsc{hook} \mid \textsc{index} & \boxed{1} \end{bmatrix} \end{bmatrix} \right\rangle \\ \textsc{c-cont} \mid \textsc{rels} & \left\langle ! \begin{bmatrix} \textsc{arg2} & \boxed{1} \end{bmatrix} ! \right\rangle \\ \textsc{dtr} \mid \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} & \langle\,\rangle \end{bmatrix}$$

(25)
$$\begin{bmatrix} \textit{added-arg2of3-lex-rule} & \\ \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} & \left\langle \begin{bmatrix} \textsc{local} & \begin{bmatrix} \textsc{cat} \mid \textsc{val} & \begin{bmatrix} \textsc{spr} & \langle\,\rangle \\ \textsc{comps} & \langle\,\rangle \end{bmatrix} \\ \textsc{cont} \mid \textsc{hook} \mid \textsc{index} & \boxed{1} \end{bmatrix} \end{bmatrix}, \boxed{2} \right\rangle \\ \textsc{c-cont} \mid \textsc{rels} & \left\langle ! \begin{bmatrix} \textsc{arg2} & \boxed{1} \end{bmatrix} ! \right\rangle \\ \textsc{dtr} \mid \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} & \langle \boxed{2} \rangle \end{bmatrix}$$

In a similar fashion, constraining the HEAD type of the added argument can be isolated to an individual rule supertype, as in (26):

(26)
$$\begin{bmatrix} \textit{added-arg2-np-head-lex-rule} \\ \textsc{synsem} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{val} \mid \textsc{comps} \mid \textsc{first} \mid \textsc{local} \mid \textsc{cat} \mid \textsc{head} \quad \textit{noun} \end{bmatrix}$$

The non-local dependencies carried by the added argument, as analyzed in the threading analysis of Bouma et al. (2001), are implemented in the Grammar Matrix as difference-list append operations. This is normally handled in the Grammar Matrix by a lexical type's mapping from argument structure to valence lists, with the additional difference-list append constraints provided via inheriting the appropriate lexical supertype (*basic-one-arg*, *basic-two-arg*, etc.). As this analysis breaks the continuity from ARG-ST, an additional constraint must be added to perform these appends.[10] An example of this operation is shown in (27):

[10] I have arbitrarily selected the ordering here of the added non-local dependencies; this analysis may need to be refined in the event that order of non-local dependency satisfaction becomes relevant.

(27)
$$\left[\begin{array}{l}\textit{added-arg2of3-non-local-lex-rule} \\ \textsc{synsem} \left[\begin{array}{ll} \textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val}\,|\,\textsc{comps}\,|\,\textsc{first}\,|\,\textsc{non-local} & \left[\begin{array}{ll}\textsc{slash} & \boxed{4} \\ \textsc{rel} & \boxed{5} \\ \textsc{que} & \boxed{6}\end{array}\right] \\ \textsc{non-local} & \left[\begin{array}{ll}\textsc{slash} & \boxed{1}\oplus\boxed{4} \\ \textsc{rel} & \boxed{2}\oplus\boxed{5} \\ \textsc{que} & \boxed{3}\oplus\boxed{6}\end{array}\right]\end{array}\right] \\ \textsc{dtr}\,|\,\textsc{synsem}\,|\,\textsc{non-local} \left[\begin{array}{ll}\textsc{slash} & \boxed{1} \\ \textsc{rel} & \boxed{2} \\ \textsc{que} & \boxed{3}\end{array}\right]\end{array}\right]$$

The most variable, individualized component is the predicate (PRED value) of the added semantic relation. For example, the benefactive and instrumental rules may be entirely common in structure, but would need to be distinguished by their predicate.[11] A separate rule supertype, therefore, can be created as in (28) to constrain the PRED value appropriately.

(28)
$$\left[\begin{array}{l}\textit{benefactive-pred-lex-rule} \\ \textsc{c-cont}\,|\,\textsc{rels} \left\langle !\left[\textsc{pred} \quad \textit{“benefactive”}\right]!\right\rangle\end{array}\right]$$

Finally, the (invariant) core of the “generic” applicative can be isolated and analyzed as in (29). The core function of this rule supertype is to contribute the new predication and link its ARG1 to the daughter’s INDEX.

(29)
$$\left[\begin{array}{l}\textit{basic-applicative-lex-rule} \\ \textsc{c-cont} \left[\begin{array}{ll}\textsc{rels} & \left\langle !\left[\begin{array}{ll}\textit{event-relation} \\ \textsc{arg1} & \boxed{1}\end{array}\right]!\right\rangle \\ \textsc{hcons} & \left\langle !\quad !\right\rangle\end{array}\right] \\ \textsc{dtr}\,|\,\textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index} \quad \boxed{1}\end{array}\right]$$

[11] This would be the case, for example, when distinct morphemes have different predicates. In the event they are not distinguishable, an underspecified predicate such as “benefactive+or+instrumental” would be more appropriate.

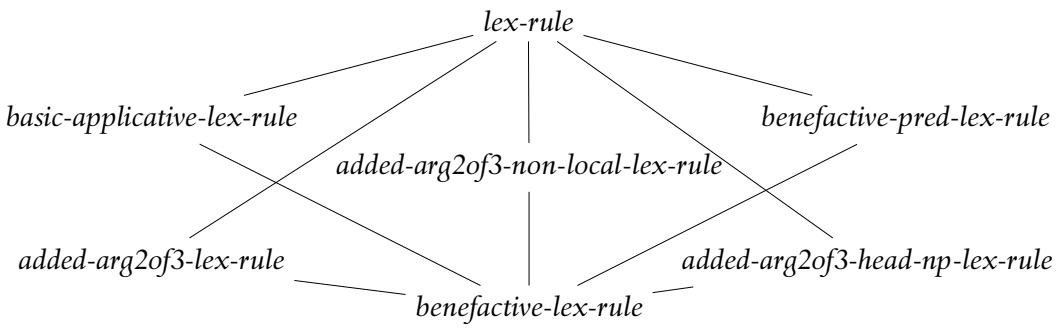


Figure 4.1: Example of rule component type hierarchy for applicative

These building-block rule component supertypes can then be assembled as inherited constraints on a complete applicative rule type, ready to be instantiated in a grammar. The partial inheritance tree showing these rule component supertypes for the example full rule type in (23) is illustrated below in Figure 4.1.

In this subsection I have developed an analysis for a full benefactive lexical rule type, shown how it can be broken down into rule components which can be parameterized on several axes of grammar-internal and cross-linguistic variation. I have also shown how these rule components can then be composed via type inheritance to produce a complete lexical rule type carrying the unified set of constraints. In the next subsection, I expand on these analyses to apply them to subject-adding operations.

4.3.2 Subject addition

The canonical subject-adding operation is the causative, which introduces a new argument into the subject role and moves the erstwhile subject into another position. In the case of an underlying intransitive, it is simply moved into object position; in the transitive H&MB give three strategies for the placement of the prior subject: (a) as an indirect object (i.e., more oblique than the other complements), (b) as an instrumental phrase, and (c) as a second direct object. (Examples of these patterns are given in Chapter 3.)

In contrast to the applicative, I treat the causative as a scopal predicate: the "causing" EP outscopes the underlying verb's EP and so provides the HOOK feature values for the entire VP. Taking the Georgian causative from Chapter 3 (repeated here as (30)) as an example, the resulting MRS should be as shown in (31):

(30) Mama-m Mzia-s daanteb-in-a cecxli
father-ERG Mzia-DAT light-CAUS-AOR:3SG fire(ABS)
'Father made Mzia light the fire.' [kat] (Harris, 1981, in H&MB, p. 12)

(31)
$$\left[\begin{array}{ll}
\textsc{hook} & \left[\begin{array}{ll}\textsc{ltop} & \boxed{5}\\ \textsc{index} & \boxed{6}\\ \textsc{xarg} & \boxed{3}\end{array}\right]\\
\textsc{rels} & \left\langle \left[\begin{array}{ll}\textbf{_daanteb_v_light} & \\ \textsc{lbl} & \boxed{8}\\ \textsc{arg}0 & \boxed{4}\ \textit{event}\\ \textsc{arg}1 & \boxed{1}\\ \textsc{arg}2 & \boxed{2}\end{array}\right], \left[\begin{array}{ll}\textbf{_mama_n_father} & \\ \textsc{arg}0 & \boxed{3}\end{array}\right], \left[\begin{array}{ll}\textbf{named} & \\ \textsc{arg}0 & \boxed{1}\end{array}\right], \left[\begin{array}{ll}\textbf{_cecxli_n_fire} & \\ \textsc{arg}0 & \boxed{2}\end{array}\right], \right.\\
 & \left. \left[\begin{array}{ll}\textbf{causative} & \\ \textsc{lbl} & \boxed{5}\\ \textsc{arg}0 & \boxed{6}\ \textit{event}\\ \textsc{arg}1 & \boxed{3}\\ \textsc{arg}2 & \boxed{1}\\ \textsc{arg}3 & \boxed{7}\end{array}\right] \right\rangle\\
\textsc{hcons} & \left\langle ! \left[\begin{array}{ll}\textit{qeq} & \\ \textsc{harg} & \boxed{7}\\ \textsc{larg} & \boxed{8}\end{array}\right] ! \right\rangle
\end{array}\right]$$

Note that, consistent with the strategy in Copestake et al. (2001), the scopal relationship is expressed by a handle constraint (HCONS) rather than directly, representing equality modulo quantifiers ($=_q$). This handle constraint predicts that quantifiers can scope in between the EPs of the causative and embedded verb.

Similarly to my analysis of the applicative, the causative can also be decomposed into component operations, again parameterized along the axes of cross-linguistic variation, as shown in

rule component	varies by
added argument	does not vary (always the subject)
constraint on new argument	constraint (e.g. case)
position of erstwhile subject	position, case, head
non-local dependencies	position (obliqueness)
HOOK feature values	do not vary
new EP's PRED value	predicate
new EP's ARG1	does not vary (always the subject)
new EP's ARG2	does not vary (always the erstwhile subject)
new EP's ARG3	does not vary

Table 4.2: Rule component axes of variation (causative)

Table 4.2.[12] The non-varying operations can be combined into a single lexical rule type to create the new relation and set its arguments, HOOK, and HCONS values, illustrated in (32):

(32)
$$\left[\begin{array}{ll}
\textit{scopal-rel-lex-rule} & \\
\textsc{c-cont} & \left[\begin{array}{ll}
\textsc{rels} & \left\langle !\left[\begin{array}{ll} \textit{event-relation} & \\ \textsc{lbl} & \boxed{5} \\ \textsc{arg0} & \boxed{6} \\ \textsc{arg1} & \boxed{3} \\ \textsc{arg2} & \boxed{1} \\ \textsc{arg3} & \boxed{7} \end{array}\right]! \right\rangle \\
\textsc{hook} & \left[\begin{array}{ll} \textsc{ltop} & \boxed{5} \\ \textsc{index} & \boxed{6} \\ \textsc{xarg} & \boxed{3} \end{array}\right] \\
\textsc{hcons} & \left\langle !\left[\begin{array}{ll} \textit{qeq} & \\ \textsc{harg} & \boxed{7} \\ \textsc{larg} & \boxed{4} \end{array}\right]! \right\rangle
\end{array}\right] \\
\textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val}\,|\,\textsc{subj} & \left\langle \left[\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index} \quad \boxed{3}\right] \right\rangle \\
\textsc{dtr}\,|\,\textsc{synsem}\,|\,\textsc{local} & \left[\begin{array}{ll} \textsc{cat}\,|\,\textsc{val}\,|\,\textsc{subj}\,|\,\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index} & \boxed{1} \\ \textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{ltop} & \boxed{4} \end{array}\right]
\end{array}\right]$$

[12] One rule component not considered in this analysis is ability to specify the HEAD value of the added subject. This capability was added to the library as a result of modeling Japanese as an illustrative language (see section 6.2.2).

The remaining position-dependent operation is where to place the erstwhile subject on the rule output's COMPS list; this rule type can, similarly to the added-argument rule type for the applicative, be defined as the output of a function:

$$g : tr \in \{intrans, trans\} \times pos \in \{front, end\} \rightarrow lrt$$

An example rule type produced by this function is shown in (33), illustrating a transitive input where the erstwhile subject is moved to the front of the COMPS list.

(33)
$$\left[\begin{array}{ll} \textit{subj-to-arg2of3-tr-op-lex-rule} & \\ \textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val} & \left[\begin{array}{ll} \textsc{subj} & \left\langle \left[\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val} \left[\begin{array}{ll} \textsc{spr} & \langle\,\rangle \\ \textsc{comps} & \langle\,\rangle \end{array}\right]\right] \right\rangle \\ \textsc{comps} & \langle \left[\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index} \quad \boxed{1}\right], \boxed{2} \rangle \end{array}\right] \\ \textsc{dtr}\,|\,\textsc{synsem}\,|\,\textsc{local}\,|\,\textsc{cat}\,|\,\textsc{val} & \left[\begin{array}{ll} \textsc{subj} & \langle \left[\textsc{local}\,|\,\textsc{cont}\,|\,\textsc{hook}\,|\,\textsc{index} \quad \boxed{1}\right] \rangle \\ \textsc{comps} & \langle \boxed{2} \rangle \end{array}\right] \end{array}\right]$$

To properly handle the non-local dependencies, the same rule constraints produced for the applicative, illustrated in (27) can be reused. Likewise, the specific predicate name for the "causative" rule can be specified by a constraint such as (28). This demonstrates reuse of the rule components not only across multiple lexical rules of the same sort, but also across broad categories of valence-changing operations.

4.4 Subject demotion

A special case of valence change is subject demotion; instead of being completely removed from the valence structure, the original subject is demoted to a complement position and replaced by a constituent formerly in complement position. This change in the syntactic relationship of arguments is not, however, accompanied by a change in the semantic roles played by the demoted subject and promoted complement. The prototypical example is the passive, as illustrated below in an example from Japanese [jpn]. Although the arguments are in different syntactic roles, both the active (34a) and passive (34b) forms should produce the same MRS, shown in (34c):

(34) a. inu ga neko wo ot-ta
dog NOM cat ACC chase-PST
'The dog chased the cat.' [jpn]

b. neko ga inu ni o-ware-ta
cat NOM dog DAT chase-PASS-PST
'The cat was chased by the dog.' [jpn] (Bender, 2013, p. 103)

c.
$$\left[\text{RELS} \left\langle \left[\begin{array}{ll}\textbf{\_inu\_n\_dog} & \\ \text{ARG0} & \boxed{1}\end{array}\right], \left[\begin{array}{ll}\textbf{\_neko\_n\_cat} & \\ \text{ARG0} & \boxed{2}\end{array}\right], \left[\begin{array}{ll}\textbf{\_ou\_v\_chase} & \\ \text{ARG0} & \textit{event} \\ \text{ARG1} & \boxed{1} \\ \text{ARG2} & \boxed{2}\end{array}\right] \right\rangle\right]$$

A notional lexical rule type to produce this output for a transitive[13] input can be defined as in (35):

(35)
$$\left[\begin{array}{ll}\textit{subj-demote-obj-promote-lex-rule} & \\ \text{SYNSEM} \mid \text{LOCAL} \mid \text{CAT} \mid \text{VAL} & \left[\begin{array}{ll}\text{SUBJ} & \left\langle \left[\begin{array}{ll}\text{LOCAL} & \left[\begin{array}{ll}\text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{1} \\ \text{CAT} \mid \text{VAL} \mid \text{HEAD} \mid \text{CASE} & \textit{nom}\end{array}\right] \\ \text{NON-LOCAL} & \boxed{3}\end{array}\right] \right\rangle \\ \text{COMPS} \mid \text{FIRST} & \left[\begin{array}{ll}\text{LOCAL} & \left[\begin{array}{ll}\text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{2} \\ \text{CAT} \mid \text{VAL} \mid \text{HEAD} \mid \text{CASE} & \textit{dat}\end{array}\right] \\ \text{NON-LOCAL} & \boxed{4}\end{array}\right]\end{array}\right] \\ \text{C-CONT} \mid \text{HOOK} \mid \text{XARG} & \boxed{1} \\ \text{DTR} \mid \text{SYNSEM} \mid \text{LOCAL} \mid \text{CAT} \mid \text{VAL} & \left[\begin{array}{ll}\text{SUBJ} & \left\langle \left[\begin{array}{ll}\text{LOCAL} \mid \text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{2} \\ \text{NON-LOCAL} & \boxed{4}\end{array}\right] \right\rangle \\ \text{COMPS} \mid \text{FIRST} & \left[\begin{array}{ll}\text{LOCAL} \mid \text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{1} \\ \text{NON-LOCAL} & \boxed{3}\end{array}\right]\end{array}\right]\end{array}\right]$$

In a similar fashion to the decomposition of rules into component operations I describe above, this rule can be split into notional *subj-demote-op-lex-rule* and *obj-promote-op-lex-rule*

[13]This example assumes that the demotion/promotion affects the first item on the COMPS list. However, this assumption was invalidated when modeling Japanese as an illustrative language (see section 6.2.2) and the implemented analyses were suitably modified and described there. Additionally, when modeling Zulu (see section 6.2.3) the subject-demoting and object-promoting rules were modified to properly handle inputs with more than one complement.

components, as well as a component applying the CASE constraints. Looking specifically at the *obj-promote-op-lex-rule* in (36) reveals one more bit of commonality: the *subj-rem-op-tr-lex-rule* in (17) can be built by combining *obj-promote-op-lex-rule* with a simple additional constraint of the rule daughter's subject to be of type *unexpressed*.

(36)
$$\left[\begin{array}{ll} \textit{obj-promote-tr-lex-rule} & \\ \text{SYNSEM} \mid \text{LOCAL} \mid \text{CAT} \mid \text{VAL} & \left[\begin{array}{ll} \text{SUBJ} & \left\langle \left[\begin{array}{ll} \text{LOCAL} \mid \text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{1} \\ \text{NON-LOCAL} & \boxed{3} \end{array}\right] \right\rangle \\ \text{COMPS} & \boxed{2} \end{array}\right] \\ \text{C-CONT} \mid \text{HOOK} \mid \text{XARG} & \boxed{1} \\ \text{DTR} \mid \text{SYNSEM} \mid \text{LOCAL} \mid \text{CAT} \mid \text{VAL} & \left[\begin{array}{ll} \text{COMPS} & \left[\begin{array}{ll} \text{FIRST} & \left[\begin{array}{ll} \text{LOCAL} \mid \text{CONT} \mid \text{HOOK} \mid \text{INDEX} & \boxed{1} \\ \text{NON-LOCAL} & \boxed{3} \end{array}\right] \\ \text{REST} & \boxed{2} \end{array}\right] \end{array}\right] \end{array}\right]$$

4.5 Summary

In this chapter, I have developed analyses of lexical rule types for five major classes of valence-changing operations: subject-removing, object-removing, subject-adding, object-adding, and subject-demoting. I have further shown how these rule types can be broken down into smaller rule types for individual component operations, which can then be recomposed into the final rule types while reusing common elements.

It is useful here to summarize the component operation rule types developed above, along with the valence-changing operations they are composed into and whether they are constant (C) or parameterized (P). This summary is shown in Table 4.3.

In the next chapter, I describe how these analysis are implemented in TDL and parameterized with respect to inputs to the customization system.

	Used in					
component rule	subj-rem	obj-rem	subj-add	obj-add	subj-dem	example(s)
subj-rem-op[a]	P					(17)(19)
obj-rem-op		C				(20)
added-argNofM				P		(24)(25)
added-argN-head				P		(26)
added-argNofM-non-local			P	P		(27)
predicate-name			P	P		(28)
basic-applicative				C		(29)
scopal-rel			C			(32)
causative-to-argNofM-tr-op			P			(33)
subj-demote-op					P	-
obj-promote-op	P				P	(36)
subj-unexpressed	C					-

[a]Can be implemented as a combination of *obj-promote-op* and *subj-unexpressed*

Table 4.3: Rule component constraint inventory

Chapter 5

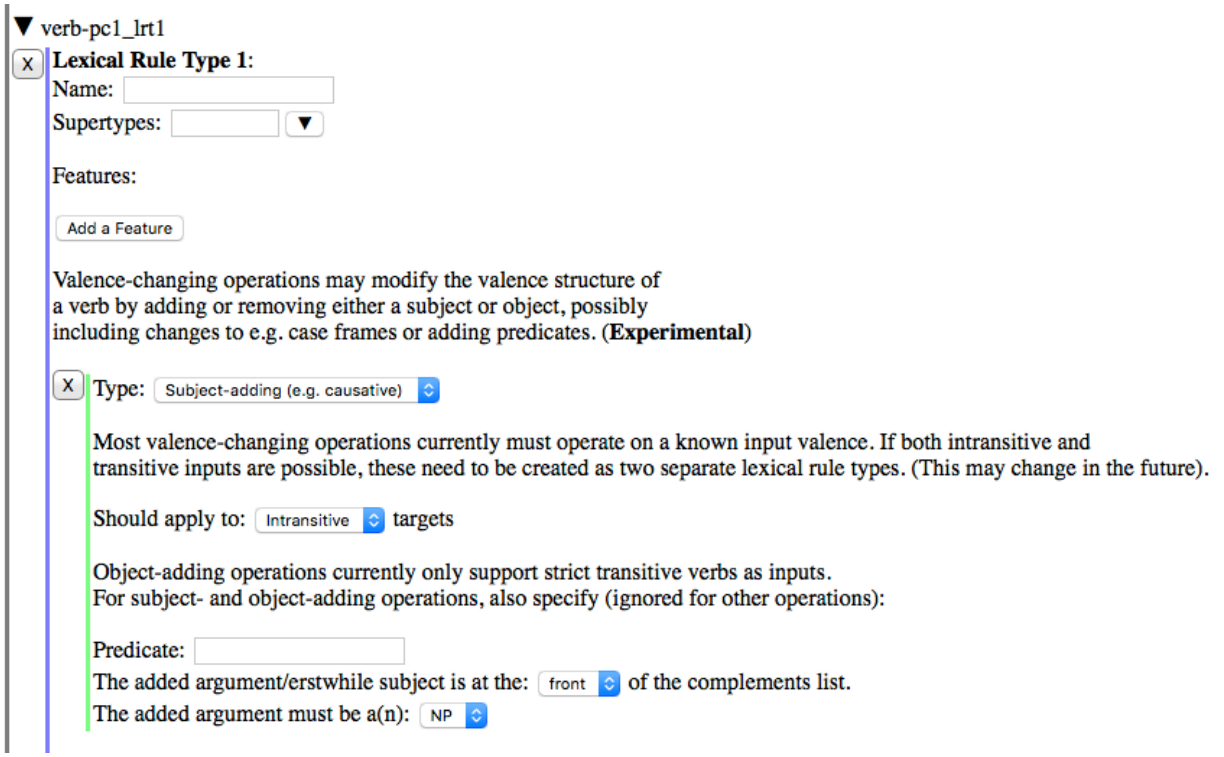
IMPLEMENTATION

In this chapter, I describe the implementation within the Grammar Matrix customization system of the scheme of lexical rule types developed in the previous chapter. I begin with the user-facing aspects and the parameters elicited via additions to the questionnaire, along with their representation in the choices file. I then describe how the library applies the inputs from the choices file to generate the TDL statements that implement the analyses, suitably parameterized by user input, as part of an internally-consistent starter grammar. As with Chapter 4, the implementation described in this section represents the implementation prior to evaluation; I have noted here where significant changes were made as a result of working with illustrative languages.

5.1 Questionnaire and user input elicitation

As described in Chapter 2, the primary interaction with the grammar writer is through the customization system questionnaire. The questionnaire is divided into subpages which, when rendered, guide the user through a process of eliciting the characteristics of phenomena in the language to be implemented in the grammar under development. The subpages provide context for each choice and may constrain available input selections to valid combinations. As the scope of this work is valence-changing verbal morphology, the new questionnaire elements are located entirely on the Morphology subpage. To define a valence-changing lexical rule, the user follows the usual process of adding a position class and lexical rule type, then clicking the "Add a valence-changing operation" button. This then adds a block for the user to select the desired

Figure 5.1: Valence-changing morphology questionnaire section



generic operation (subject-adding, object-removing, etc.) and the parameters needed to specify the behavior.[1] This section of the questionnaire is show in figure 5.1.

The selection lists and text inputs in this part of the questionnaire map to the parameterization axes for the component rules described in the previous chapter. The mapping is generally transparent as to what a particular parameter will affect in the generated TDL; however, one simplification is the option for selecting whether an added argument (or the erstwhile subject) is to be added to the front or end of the COMPS list. As I show later in this chapter, this option is used, along with whether the operation selects for transitive or intransitive targets, to generate rules specific to the expected valence patterns (*e.g.* rules of the form 'argNofM').

[1] Due to limitations in the current questionnaire implementation, the set of possible parameters presented to the user is not specific to the selected operation; parameters not used by the selected operation are ignored. This will be addressed in future work.

5.2 Type definitions and TDL generation

In this section, I describe the strategy I developed for generating parameterized lexical rule types, the set of lexical rule types used to implement the valence-changing operations, and the specific implementations of these rules.

5.2.1 Library structure and generation strategy

As described in Chapter 2, a "library" in the Grammar Matrix customization system refers to a set of analyses, elements of a questionnaire, and the code necessary to convert the questionnaire inputs into an appropriately-structured and -integrated set of TDL implementations of the analyses. The building-block approach I developed in this work relies on the combination of multiple rule supertypes, each of which implements a specific operation, with a specified configuration and feature values as determined by input parameters; formally, given the set **P** of input parameters p, each lexical rule type is the output of some generating function $f_i \in \mathbf{F} : p_1 \times \cdots \times p_n \rightarrow lrt$.

In my implementation of this library, this structure is expressed directly: for each of the parameterizable lexical rule types (corresponding to a component operation), a generating function exists that produces the correct TDL fragment implementing the variant of that rule type appropriate to the input parameter values. These component operation building-block rules are combined to produce the requested valence-changing operation(s); any given resulting grammar will, therefore, likely contain multiple rule types expressing the realized variants of those component operations and their allowable combinations. As the ultimate assembly and instantiation of the combined operations as rule instances attached to position classes is handled by the morphotactics library, I used this structure to maintain a degree of modularity and keep the valence change-specific logic localized as much as possible to the valence change library proper.[2]

[2]Concretely, the TDL-generating logic is contained in the `valence_change.py` module and exposed to the `morphotactics.py` module via explicit imports to avoid creating circular dependencies.

5.2.2 Lexical rule types

In this section, I detail each of the individual component operation rule types called out in the analysis of Chapter 4, along with its corresponding TDL fragments. In particular, I highlight the parameterized elements corresponding to the variants produced, along with any particular implementation concerns.

subj-rem-op

The rule implementing the subject-removing operation has two variants: `subj-rem-itr-op-lex-rule` and `subj-rem-tr-op-lex-rule`, corresponding to the variants selecting for intransitive and transitive targets, respectively. These variants are sufficiently different that they are generated using separate templates.

The intransitive variant, shown in (37) is straightforward, constraining the rule daughter's COMPS list to be empty (to ensure an intransitive input), and constrains its own SUBJ and COMPS lists to be empty. It also constrains the daughter SUBJ to be of type `unexpressed`; as noted in Chapter 3, this special type serves to ensure that the non-local dependency lists (SLASH, QUE, and REL) are properly grounded in empty difference lists.

(37)
```
subj-rem-itr-op-rule := subj-change-only-lex-rule &&
    [ SYNSEM.LOCAL.CAT.VAL [ SUBJ < >,
                             COMPS < > ],
      DTR.SYNSEM.LOCAL.CAT.VAL [ SUBJ < unexpressed >,
                                 COMPS < > ] ].
```

The transitive variant in (38) is more complex by comparison, as it coindexes the output SUBJ with the first item on the input COMPS list while also exposing it as the rule output's XARG. (See example (17) in Chapter 4 for the details of this analysis.) This rule also preserves the NON-LOCAL features on the argument; to allow the HEAD type constraint to be altered (*e.g.*, to allow a change from NP to PP) only the MOD and KEYS values of the HEAD are copied.[3]

[3]This implementation of subject removal unnecessarily entails the promotion of a complement into subject position. A future revision should separate pure subject removal (marking the daughter's SUBJ as *unexpressed*) from

(38)
```
subj-rem-tr-op-rule := subj-and-comps-change-only-lex-rule &
  [ SYNSEM [ LOCAL.CAT.VAL [ SUBJ < [ LOCAL [ CONT.HOOK.INDEX #sind,
                                              CAT [ HEAD [ MOD #mod,
                                                           KEYS #keys ],
                                                    VAL #val ] ],
                                         NON-LOCAL #nl ] >,
                              COMPS #rest ] ],
    C-CONT.HOOK.XARG #sind,
    DTR.SYNSEM.LOCAL [ CAT.VAL [ SUBJ < unexpressed >,
                                 COMPS [ FIRST [ LOCAL [ CONT.HOOK.INDEX #sind,
                                                         CAT [ HEAD [ MOD #mod,
                                                                      KEYS #keys ],
                                                               VAL #val ] ],
                                                   NON-LOCAL #nl ],
                                         REST #rest ] ] ] ].
```

subj-dem-op

The subject-*demoting* rule differs from subject removal in that the argument is not actually removed (constrained to be *unexpressed* in the daughter). Instead, the input SUBJ is coindexed at the front of the output COMPS list,[4] without constraining the output SUBJ. This rule is illustrated in (39) below:

(39)
```
subj-dem-op-lex-rule := subj-and-comps-change-only-lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL.COMPS < [ LOCAL [ CAT.VAL [ SPR #spr,
                                                     COMPS #comps ],
                                           CONT.HOOK.INDEX #sidx ],
                                 NON-LOCAL #nl ],
                                #oarg >,
    DTR.SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL [ CAT.VAL [ SPR #spr,
                                                          COMPS #comps ],
                                                CONT.HOOK.INDEX #sidx ],
                                       NON-LOCAL #nl ] >,
                               COMPS < #oarg > ] ].
```

---

object promotion; this would remove the need for transitivity-specific variants. By reusing the object promotion rule, this approach would further embrace the modular approach I developed for this library.

[4] As described in section 4.4, the subject-demoting and object-promoting rules were parameterized to target different COMPS positions as a result of modeling Japanese (see section 6.2.2) and modified to properly handle multiple input complements as a result of modeling Zulu (section 6.2.3).

obj-rem-op

The object-removing rule type, shown in (40) is quite simple; it ignores the first item[5] on the input COMPS list and "promotes" the rest of the input COMPS list to the output COMPS. Note that again the special type `unexpressed` is applied as a constraint to the "deleted" argument to maintain the proper handling of non-local dependencies.

(40)
```
obj-rem-op-rule := comps-change-only-lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL.COMPS #comps,
    DTR.SYNSEM.LOCAL.CAT.VAL.COMPS < unexpressed . #comps > ].
```

obj-prom-op

Rather than removing an object completely, the object-promoting rule type, shown in (41), coindexes the first[6] complement on the COMPS list with the output SUBJ.

(41)
```
obj-prom-op-lex-rule := subj-and-comps-change-only-lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL [ CAT.VAL [ SPR #spr,
                                                      COMPS #comps ],
                                            CONT.HOOK.INDEX #sidx ],
                                     NON-LOCAL #nl ] >,
                           COMPS < [ ],
                                   #oarg > ],
    DTR.SYNSEM.LOCAL.CAT.VAL.COMPS < [ LOCAL [ CAT.VAL [ SPR #spr,
                                                         COMPS #comps ],
                                               CONT.HOOK.INDEX #sidx ],
                                        NON-LOCAL #nl ],
                                   #oarg > ].
```

basic-applicative

The `basic-applicative` rule type is used for all applicative valence-changing operations; it adds a new elementary predication to the output semantics RELS list. This EP has as its ARG1 the

[5] My work modeling Zulu as an illustrative language (see section 6.2.3) revealed that targeting only the first item was an invalid assumption, and the rule was modified to vary based on the position of the target complement.

[6] See footnote 4 in this chapter.

HOOK.INDEX of the rule input (*i.e.*, the event ARG0 of the underlying verb). As discussed in Chapter 4, in my analysis the added predication is non-scopal and so its arguments do not have handle relationships interposed. This rule, shown in (42), is also not parameterized.

(42)
```
basic-applicative-lex-rule := comps-change-only-lex-rule &
    [ C-CONT [ RELS <! event-relation &
                       [ ARG1 #idx ] !>,
               HCONS <! !> ],
      DTR.SYNSEM.LOCAL.CONT.HOOK.INDEX #idx ].
```

added-argNofM-applicative

The `added-argNofM-applicative` rule type is parameterized on the position of the added argument (`N`) and the total number of arguments in the rule output (`M`). In this implementation, both N and M are in the set {2, 3} for the applicative.[7] This rule type inherits from the `basic-applicative-lex-rule` supertype above, as well as from the correspondingly-parameterized `added-argNofM-non-local-lex-rule` described below. This rule performs two primary functions: it adds the new argument in the appropriate position on the output COMPS list and coindexes that argument with the ARG2 of the new EP (added by `basic-applicative-lex-rule`). Both an intransitive and transitive variant are illustrated here for comparison in example (43); the key variably-positioned element—the rule daughter's COMPS—is shown highlighted in **`bold`**. Note also that the non-local features of the added argument are handled here by inheriting from the `added-argNofM-non-local` rule type (described in the next section).

(43) a.
```
added-arg2of2-applicative-lex-rule := basic-applicative-rule &
                                      added-arg2of2-non-local-lex-rule &
    [ SYNSEM.LOCAL.CAT.VAL.COMPS < [ LOCAL [ CAT [ VAL [ SPR < >,
                                                         COMPS < > ] ],
                                           CONT.HOOK.INDEX #nind ] ] >,
      C-CONT.RELS <! [ ARG2 #nind ] !>,
      DTR.SYNSEM.LOCAL.CAT.VAL.COMPS < > ].
```

[7] The customization system cannot directly create ditransitive verbs, so the input can only have one or two arguments to add to.

b.
```
added-arg2of3-applicative-lex-rule := basic-applicative-rule &
                                      added-arg2of3-non-local-lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL.COMPS < [ LOCAL [ CAT [ VAL [ SPR < >,
                                                       COMPS < > ] ],
                                               CONT.HOOK.INDEX #nind ] ],
                                  #ocomp >,
    C-CONT.RELS <! [ ARG2 #nind ] !>,
    DTR.SYNSEM.LOCAL.CAT.VAL.COMPS < #ocomp > ].
```

added-argNofM-non-local

Operations that add an argument need to ensure that any non-local dependencies are properly threaded into output non-local dependencies lists (SLASH, REL, and QUE), as discussed in Chapter 4. This is achieved by inheriting from the appropriate variant of the `added-argNofM-non-local` supertype, shown in (44). To conserve space, only the rules for an added subject and one variation of an added complement are shown.

(44) a.
```
added-arg1ofN-non-local-lex-rule := lex-rule &
  [ SYNSEM [ LOCAL.CAT.VAL.SUBJ < [ NON-LOCAL [ SLASH [ LIST #smiddle,
                                                        LAST #slast ],
                                                 REL [ LIST #rmiddle,
                                                       LAST #rlast ],
                                                 QUE [ LIST #qmiddle,
                                                       LAST #qlast ] ] ] >,
             NON-LOCAL [ SLASH [ LIST #sfirst,
                                 LAST #slast ],
                         REL [ LIST #rfirst,
                               LAST #rlast ],
                         QUE [ LIST #qfirst,
                               LAST #qlast ] ] ],
    DTR.SYNSEM.NON-LOCAL [ SLASH [ LIST #sfirst,
                                   LAST #smiddle ],
                           REL [ LIST #rfirst,
                                 LAST #rmiddle ],
                           QUE [ LIST #qfirst,
                                 LAST #qmiddle ] ] ].
```

b.
```
added-arg2of3-non-local-lex-rule := lex-rule &
    [ SYNSEM [ LOCAL.CAT.VAL.COMPS < [ NON-LOCAL [ SLASH [ LIST #smiddle,
                                                           LAST #slast ],
                                                    REL [ LIST #rmiddle,
                                                          LAST #rlast ],
                                                    QUE [ LIST #qmiddle,
                                                          LAST #qlast ] ] ],
                                       [ ] >,
                     NON-LOCAL [ SLASH [ LIST #sfirst,
                                         LAST #slast ],
                                 REL [ LIST #rfirst,
                                       LAST #rlast ],
                                 QUE [ LIST #qfirst,
                                       LAST #qlast ] ] ],
      DTR.SYNSEM.NON-LOCAL [ SLASH [ LIST #sfirst,
                                           LAST #smiddle ],
                             REL [ LIST #rfirst,
                                   LAST #rmiddle ],
                             QUE [ LIST #qfirst,
                                   LAST #qmiddle ] ] ].
```

basic-scopal-rel

As described in Chapter 4, in my analysis of the subject-adding operations the additional EP outscopes the underlying verb's EP, with the newly-added subject as its ARG1 (the "causer"), and provides the new HOOK (including the LTOP, INDEX, and XARG). This EP is contributed, with these features appropriately constrained, by the `basic-scopal-rel` rule type, shown in (45). As with the `basic-applicative` rule type, this type is not parameterized.

(45)
```
basic-scopal-rel-lex-rule := lex-rule
  [ C-CONT [ RELS <! event-relation &
                     [ LBL #ltop,
                       ARG0 #hidx,
                       ARG1 #causer ] !>,
             HOOK [ LTOP #ltop,
                    INDEX #hidx,
                    XARG #causer ] ],
    SYNSEM.LOCAL.CAT.VAL.SUBJ < [ LOCAL.CONT.HOOK.INDEX #causer ] > ].
```

scopal-N-place-rel

The variation in the scopal relation added by subject-adding operations is expressed in the `scopal-N-place-rel` rule types. In the two-place variant (46a), the outscoped ("caused") event, *i.e.*, the rule daughter's LTOP, is connected to the ARG2 via a handle constraint added to HCONS. By contrast, in the three-place variant (46b), the erstwhile subject (the rule daughter's XARG) is identified with ARG2 and the outscoped event is connected (via a handle constraint) to ARG3. Providing both variants allows the causative to be modeled as either a raising or control predicate.

(46) a.
```
scopal-2-place-rel-lex-rule := basic-scopal-rel-lex-rule &
  [ C-CONT [ RELS <! [ ARG2 #chandle ] !>,
             HCONS <! [ qeq &
                        [ HARG #chandle,
                          LARG #caused ] ] !> ],
    DTR.SYNSEM.LOCAL.CONT.HOOK.LTOP #caused ].
```
b.
```
scopal-3-place-rel-lex-rule := basic-scopal-rel-lex-rule &
  [ C-CONT [ RELS <! [ ARG2 #causee,
                       ARG3 #chandle ] !>,
             HCONS <! [ qeq &
                        [ HARG #chandle,
                          LARG #caused ] ] !> ],
    DTR.SYNSEM.LOCAL [ CAT.VAL.SUBJ < [ LOCAL.CONT.HOOK.INDEX #causee ] >,
                       CONT.HOOK.LTOP #caused ] ].
```

causative-to-argNofM-tr-op

Analogously to the `added-argNofM-applicative` rule types, subject-adding operations also need to constrain where on the COMPS list the erstwhile subject should be moved. Naming of these types is slightly different, however; instead of the `argNofM` element representing the position of the new argument, here `argNofM` refers to the position of the old subject. To make this explicit, these rule type names are of the form `causative-`**`to`**`-argNofM-op`. Examples of the variants for intransitive and transitive are shown in (47).

(47) a.
```
causative-to-arg2of2-op-lex-rule := lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL.CAT.VAL [ SPR < >,
                                                    COMPS < > ] ] >,
                           COMPS < [ LOCAL [ CONT.HOOK.INDEX #arg2,
                                             CAT [ HEAD [ MOD #mod,
                                                          KEYS #keys ],
                                                   VAL #val ] ],
                                     NON-LOCAL #nl ] > ],
    DTR.SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL [ CONT.HOOK.INDEX #arg2,
                                                CAT [ HEAD [ MOD #mod,
                                                             KEYS #keys ],
                                                      VAL #val ] ],
                                       NON-LOCAL #nl ] >,
                               COMPS < > ] ].
```

b.
```
causative-to-arg3of3-op-lex-rule := lex-rule &
  [ SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL.CAT.VAL [ SPR < >,
                                                    COMPS < > ] ] >,
                           COMPS < #comp,
                                   [ LOCAL [ CONT.HOOK.INDEX #arg2,
                                             CAT [ HEAD [ MOD #mod,
                                                          KEYS #keys ],
                                                   VAL #val ] ],
                                     NON-LOCAL #nl ] > ],
    DTR.SYNSEM.LOCAL.CAT.VAL [ SUBJ < [ LOCAL [ CONT.HOOK.INDEX #arg2,
                                                CAT [ HEAD [ MOD #mod,
                                                             KEYS #keys ],
                                                      VAL #val ] ],
                                       NON-LOCAL #nl ] >,
                               COMPS < #comp > ] ].
```

predicate-name

Operations that add new elementary predications also need to specify the PRED value for the EPs they add. As this EP represents the specific contribution to the semantics of the valence-changing operation itself, the PRED value, in describing the relation, is the element most specific to the valence-changing operation being constructed. The PRED value is thus added to the rule type corresponding to the high-level operation, *i.e.*, the rule type intended to be directly instantiated. Since each argument-adding operation only adds a single EP, and does so via the C-CONT

list, the predicate name of the relation can be attached via a simple constraint, as in the notional rule type[8] in (48).

(48)
```
causative-lex-rule := lex-rule &
  [ C-CONT.RELS <! [ PRED "causative" ] !> ].
```

5.2.3 Lexical rule type assembly

In the previous section I illustrated how the rule types that implement the component operation "building blocks" are created, with variants determined by user inputs as parameters (as well as several constant rule types). In this section I show how the building blocks are assembled into the rule types that implement the full valence change operations, ready to be instantiated as rules.

To illustrate this assembly process, take a hypothetical language with a three-place causative expressed by a verbal affix on underlyingly transitive verbs. In this notional causative, the erstwhile subject becomes the least-oblique complement of the underlying verb (*i.e.*, is at the front of the COMPS list) and the grammar creator has determined that the added predication should have a PRED value of "causative". The new subject must be a noun phrase, and the rule type itself should be called `causative-tr-lex-rule`. This rule type would be created in the questionnaire by adding a valence-changing operation to a lexical rule type, with the following inputs:

- **Subject-adding** operation
- Should apply to **transitive** targets
- Erstwhile subject is at the **front** of the complements list
- The added argument must be **an NP**
- The added EP should have a predicate name of **"causative"**

With these inputs, the following rule type variants (described above) will be generated:

- `basic-scopal-rel-lex-rule`
- `scopal-3-place-rel-lex-rule`
- `causative-to-arg2of3-tr-op-lex-rule`

[8] The actual supertypes of this rule type are more complex, as this type must inherit all the constraints that implement the operation. (Details of how this rule type is constructed follow in Section 5.2.3.)

- `added-arg1of3-np-head-lex-rule`
- `causative-tr-lex-rule`

Note that this last rule, `causative-tr-lex-rule`, is the rule name the user specified in the questionnaire and is the rule type that will be instantiated by the rule instances with their associated morphology. This rule type combines the rest of the constraint rule types to achieve the complete causative operation. Because nearly all the constraints come from building block rule types, the actual rule type definition itself is quite simple, as shown in (49) below:

(49)
```
causative-tr-lex-rule := basic-scopal-rel-lex-rule &
                         scopal-3-place-rel-lex-rule &
                         causative-to-arg2of3-tr-op-lex-rule &
                         added-arg1of3-np-head-lex-rule &
    [ C-CONT.RELS <! [ PRED "causative" ] !> ].
```

As described above, the predicate name is directly contributed by this comprehensive rule type, which allows the building-block rule types to remain more generic and reusable; for example, for affixes that have exactly the same syntactic effects, but contrasting semantics. This technique of assembling the "user-facing" rules through inheritance is followed for all valence-changing operations supported in this library: subject-adding, subject-removing, object-adding, object-removing, and subject-demoting.

5.3 Summary

In this chapter I have described the implementation of the lexical rule types developed in my analyses, including the user questionnaire for parameter elicitation and the TDL generation mechanisms, and illustrated the variations of implementations generated by my library based on input parameters. In the next chapter, I discuss my evaluation of the library first using a set of constructed pseudolanguages, and then several illustrative languages used during development. Finally, I evaluate the library against a number of held-out languages to assess the degree to which my approach and the building blocks created support valence-changing operations cross-linguistically.

Chapter 6
EVALUATION

In this chapter, I discuss several levels of evaluation of the library. First, I describe the collection of pseudolanguage regression tests I wrote and used to guide and monitor development. Next, I discuss several examples of phenomena from natural languages selected to illustrate key capabilities of the library. Finally, I present an evaluation of the library against five additional natural languages that I held out during development.

6.1 Regression tests

The Grammar Matrix customization system includes a suite of regression tests, created by developers as they add to either the Grammar Matrix core, customization system, or customization system libraries (Bender et al., 2007). These regression tests provide an automated means for (re)validating that the modified system, as a whole, behaves in the same way as it did prior to modification, modulo expected new behavior (Agrawal, Horgan, Krauser, & London, 1993). Notably, developers are encouraged to write these tests *before* beginning coding, applying the principle of Test-Driven Development (Beck, 2002): a test is written that describes the expected output, which initially fails; code is subsequently written to make the test pass. From a software engineering perspective,[1] this accomplishes the dual goals of reducing software defects (Williams, Maximilien, & Vouk, 2003) and creating regression tests for future use; after all, the new system will be the "old" system when later modifications are made. From a linguistic perspective, the regression tests are especially helpful in articulating precisely the phenomena to be implemented, with examples of both positive and negative grammaticality. As the complete

[1]There is a substantial literature on both regression testing and test-driven development that is not relevant to the present work; I merely limn the principles here to show that the library development process is grounded in sound software engineering practices.

range of possibilities and configuration space to be covered is not fully known at the outset, however, development of regression tests is expected to continue during the coding phase of library development.

A regression test includes a test suite of sentences marked with expected grammaticality, a choices file for the customization system that specifies a grammar that should parse the grammatical sentences in the test suite (and not parse those marked as ungrammatical), and gold-standard semantic outputs for the sentences that do parse. The earliest tests written in the process of developing a library, generally making up the bulk of the tests,[2] are written using pseudolanguages; that is, they express syntactic phenomena selected to exercise certain system behaviors, but using a simplistic lexicon chosen for developer convenience and clarity, rather than attempting to reflect any specific natural language. An example of such a pseudolanguage test suite is shown below in (50); this example illustrates the subject-removing operation for SVO word order:[3]

(50)
```
*iv                        *tv
iv-nosubjitr               *tv-nosubjitr
*iv-nosubjtr               *tv-nosubjtr
n1 iv                      *n2 tv
*n1 iv-nosubjitr           *n2 tv-nosubjitr
*n1 iv-nosubjtr            n2 tv-nosubjtr
*n1 iv n2                  n1 tv n2
*n1 iv-nosubjitr n2        *n1 tv-nosubjitr n2
*n1 iv-nosubjtr n2         *n1 tv-nosubjtr n2
```

A leading asterisk marks a test suite sentence as ungrammatical and not expected to parse. For instance, in the above example the test sentence `*n1 iv-nosubjitr` is intended to verify that the created grammar will not parse a sentence with an overt NP (`n1`) followed by an intransitive verb (`iv`) with the intransitive form of the subject-removing affix (`-nosubjitr`). As can be seen, the pseudolanguage lexicon simply reflects the syntactic role of each morpheme.

---

[2]The test suites for illustrative and held-out languages are typically also added as regression tests at the conclusion of library development.

[3]This test suite is shown here in two columns for space reasons; the actual file is structured with one sentence per line.

A fragment of the choices file that accompanies this particular test is shown in (51) below; I have edited the presentation to include only the elements that specifically exercise the new functionality provided by my library. The omitted sections establish SVO word order, that this pseudolanguage does not have determiners or auxiliaries, and that it does not mark for person or case; they also define lexical entries for two distinct nouns and instances of both intransitive and transitive verbs. The elided sections all use the general existing customization system; the new elements shown here illustrate additional capability added to the `morphology` section by my valence change library. These additions, by defining lexical rule types with valence-changing operations, direct the library to create and assemble the appropriate component operation and combined rule types and rule instances to implement the subject-removing operation.

(51)
```
section=morphology
  verb-pc1_order=suffix
  verb-pc1_inputs=verb
    verb-pc1_lrt1_name=subjrem-itr
      verb-pc1_lrt1_valchg1_operation=subj-rem
      verb-pc1_lrt1_valchg1_inputs=intrans
      verb-pc1_lrt1_lri1_inflecting=yes
      verb-pc1_lrt1_lri1_orth=-nosubjitr
    verb-pc1_lrt2_name=subjrem-tr
      verb-pc1_lrt2_valchg1_operation=subj-rem
      verb-pc1_lrt2_valchg1_inputs=trans
      verb-pc1_lrt2_lri1_inflecting=yes
      verb-pc1_lrt2_lri1_orth=-nosubjtr
```

The third component of this test, the set of gold-standard semantic representations expected as outputs for the grammatical sentences, is created during the development process by manually inspecting the output for each sentence and confirming that the representation is correct. The semantic representations themselves, in the form of expected MRSes, are stored in the `[incr tsdb()]` database format (Oepen & Flickinger, 1998) and are not meant to be directly interpreted outside of that system; I therefore have not reproduced them here. For a human-readable example of such a stored MRS, however, see (60) below.

		examples		performance			
Group	pseudo-languages	positives	negatives	parses	coverage	overgeneration	spurious ambiguity
Subject-removing	3	12	42	12	100%	0%	0%
Object-removing	3	6	7	6	100%	0%	0%
Subject-adding	4	16	10	16	100%	0%	0%
Object-adding	4	8	4	8	100%	0%	0%
Combinations	3	21	25	21	100%	0%	0%
Total	17	63	88	63	100%	0%	0%

Table 6.1: Pseudolanguage test summary and performance

Similarly to the test described above, I created additional pseudolanguage tests to exercise each supported valence-changing operation, with different word orders and parameter variations permitted by the library. Each of these tests is identified in the regression test framework by its pseudolanguage name, which always starts with `valchg-` for this library. An overview of the pseudolanguage test suite coverage and results are shown in Table 6.1.

6.2 Illustrative languages

In addition to the pseudolanguages used in development and testing, I also validated the approach and implementation using several human languages, from different language families, that exhibit valence-changing verbal morphology of different kinds. In this section, I give examples of relevant phenomena from each of these languages that illustrate the application of the library; the test suites for each of these languages are intentionally kept small to focus solely on valence change and avoid testing unrelated aspects of the customization system. I also describe improvements made to the system that were inspired by work on the illustrative languages, including changes necessary to cover instances of valence change in each language. I then give the performance results on the illustrative languages in Section 6.2.4.

6.2.1 Lakota

Lakota [lkt] is a critically-endangered (Moseley, 2010) language in the Siouan family, spoken by approximately 2,300 speakers (in all dialects) in the United States and Canada (Lewis, Simons, & Fenning, 2013). Lakota has complex verbal morphology, with many productive affixes and several valence-changing operations. The affix system includes both prefixing and infixing; many affixes can be prefixing or infixing, depending on the co-occurring pronominal affixes. Such non-concatenative morphosyntax exceeds the capabilities of the Grammar Matrix and customization system; I have therefore transformed my test suite sentences into a regularized, purely concatenative form, such as would be produced at the upper level of a morphophonological analyzer. (See also *e.g.* Bender and Good 2005 for additional arguments for maintaining this separation.) For an example of such an analyzer implemented for Lakota, see Curtis 2014. In the examples in this section I give the sentences in the following order:

1. surface form, in the standard orthography
2. segmented, phonologically-regularized morphemes
3. output of notional morphophonological analyzer[4]
4. morpheme-by-morpheme glosses
5. free translation

Subject removal

There is an impersonalizing construction or "agent impersonalizer" (glossed as AGIPS)[5] (Pustet & Rood, 2008) in Lakota that suppresses reference to the transitive subject, as illustrated in (52):

---

[4]Standard Lakota orthography includes two characters (ǧ and ȟ) that, although supported by Unicode, fall outside the standard Latin Extended character sets; for ease of editing I have replaced these in the input strings in the test suites and the grammar with G and H, respectively.

[5]Pustet and Rood consistently refer to *syntactic* valence slots by their typical *semantic* roles as here; for consistency with the existing literature I use their terminology and glosses in this discussion.

(52) a. waštúŋkala wakáǧe
waštúŋkala wa-káǧA
waštúŋkala 1SgAgt-káGA
dried.corn 1SG.A-make
'I made dried corn.' [lkt] (after Pustet & Rood, 2008, p. 336)

b. waštúŋkala káǧapi
waštúŋkala káǧA-pi
waštúŋkala AGIPS-káGA
dried.corn AGIPS-make
'Dried corn was made.' [lkt] (after Pustet & Rood, 2008, p. 336)

In this example, note that I have assumed a notional morphophonological analyzer output that places the agent impersonalizer in the same position class as pronominal affixes. This approach merely reduced the complications of position class interactions for the purposes of this evaluation and would not be desirable in a more comprehensive grammar.

To implement this subject-removing affix for Lakota, the relevant section of the choices file is shown below in (53). In this example, the elided lexical rule type `verb-pc1_lrt1` implements the pronominal affix.

(53)
```
section=morphology
  verb-pc1_order=prefix
  verb-pc1_inputs=verb-pc4
    verb-pc1_lrt2_name=agips-tr-fill
      verb-pc1_lrt2_valchg1_operation=subj-rem
      verb-pc1_lrt2_valchg1_inputs=trans
      verb-pc1_lrt2_lri1_inflecting=yes
      verb-pc1_lrt2_lri1_orth=AGIPS-
```

Object removal

Lakota also has an extremely productive deobjective affix *wa-*, which is variably described as an "indefinite object marker" (Ullrich, 2011; Ullrich & Black Bear, 2016), "indefinite object prefix" (Van Valin, 1977), or "inanimate[6] patient dereferentializer" or "patient impersonalizer" (glossed

[6]Pustet and Rood describe an animacy contrast here that is contradicted by *e.g.*, Ullrich and Black Bear. As the animacy axis is not relevant to this work, I make no attempt to address it.

as PATIPS) (Pustet & Rood, 2008). It applies to transitive verbs and blocks an object valence slot; for examples, see (54) below. The fragment of the choices file that controls the relevant valence-changing library additions follows at (55); as with the subject-removing affix above, I assumed the morphophonological analyzer output placed the object-removing affix in the same position class as the pronominal affix.

(54) a. šúŋka kiŋ Phita yaȟtáke šni
šúŋka kiŋ Phita yaȟtákA šni
šúŋka kiŋ Phita yaHtákA šni
dog DEF Peter bite NEG
'The dog did not bite Peter.' [lkt] (Ullrich & Black Bear, 2016, p. 200)

b. šúŋka kiŋ wayáȟtáke šni
šúŋka kiŋ wa-yaȟtáke šni
šúŋka kiŋ PATIPS-yaHtáke šni
dog DEF PATIPS-bite NEG
'The dog doesn't bite.' [lkt] (Ullrich & Black Bear, 2016, p. 200)

(55)
```
section=morphology
  verb-pc4_name=pat-obj-pron-pos
  verb-pc4_order=prefix
  verb-pc4_inputs=verb1
    verb-pc4_lrt2_name=patips-obj-fill
      verb-pc4_lrt2_valchg1_operation=obj-rem
      verb-pc4_lrt2_valchg1_inputs=trans
      verb-pc4_lrt2_lri1_inflecting=yes
      verb-pc4_lrt2_lri1_orth=PATIPS-
```

Subject addition

The Lakota causative affix *-yA* is quite productive and can be applied to stative, intransitive, and transitive verbs (as well as many nouns). A typical example from the test suite is shown in (56) below. Note especially in (56b) the added subject pronominal affix (*-wa-*) infixing between the stem and causative morpheme in the surface string; the causative therefore attaches outside the agreement marker for the added subject. This interesting property of Lakota is surprising, given

that the causative provides the subject that the agreement marker refers to. Since we treat agreement as semantically contentful, semantic compositionality requires that this be modeled as a strictly morphophonological process. Here again I leverage a notional morphophonological analyzer to convert the input into a suitably-structured form, as well as to regularize infixed stems.

(56) a. inič<u>há</u>ğe
i-ni-čháğA
2SgPat-ičháGA
2SG.P-grow
'You grew' [lkt] (Ullrich & Black Bear, 2016, p. 156)

b. inič<u>há</u>ȟwaye
i-ni-čháğA-wa-yA
1SgAgt-2SgPat-ičháGA-CAUS
1SG.A-2SG.P-grow-CAUS
'I raised you (lit. made you grow).' [lkt] (Ullrich & Black Bear, 2016, p. 156)

The relevant portion of the corresponding choices file that implements this form of the causative is shown in (57). Because this form acts on intransitive verbs, note that the choices file (and the corresponding questionnaire entry) specify `intrans` input. While this may appear redundant, given that the morphology library allows input lexical type and lexical rule type restrictions, this in fact reflects a limitation in the current implementation of my library: generated valence-changing operations are specific to one input valence structure and so need to be specified for the exact valence structure expected to appear at the input.

(57)
```
verb-pc6_name=caus-itr-pos
verb-pc6_order=suffix
verb-pc6_inputs=verb-pc5
verb-pc6_require1_others=verb-pc2
  verb-pc6_lrt1_name=caus
    verb-pc6_lrt1_valchg1_operation=subj-add
    verb-pc6_lrt1_valchg1_inputs=intrans
    verb-pc6_lrt1_valchg1_predname=caus_v_rel
    verb-pc6_lrt1_valchg1_argpos=pre
    verb-pc6_lrt1_valchg1_argtype=np
    verb-pc6_lrt1_lri1_inflecting=yes
    verb-pc6_lrt1_lri1_orth=-CAUS
```

Note also that the interpretation of choices file inputs `argpos` and `argtype` are specific to each valence-changing operation. In this instance, for example, `argpos` specifies where the erstwhile subject should be placed on the COMPS list, while `argtype` adds a constraint to the head of the added (new) subject. In other words, these choices have effects on different arguments. This differs from how these choices are interpreted for object-adding operations, as shown in the next section. The varying roles of these choices are described for the user in the questionnaire.

Object addition

The general-purpose dative[7] affix *ki-* in Lakota most frequently serves to express the benefactive (or malefactive, depending on context). It adds an object valence slot for a third argument, as shown in (58) here:

(58) a. até čháŋčheğa waŋ káğe
até čháŋčheğa waŋ káğA
até čháŋčheGa waŋ káGA
father drum INDEF make
'(my) father made a drum.' [lkt] (Ullrich & Black Bear, 2016, p. 170)

b. até hokšíla kiŋ čháŋčheğa waŋ kikáğe
até hokšíla kiŋ čháŋčheğa waŋ ki-káğA
até hokšíla kiŋ čháŋčheGa waŋ BEN-káGA
father boy DEF drum INDEF BEN-make
'(my) father made the boy a drum.' [lkt] (Ullrich & Black Bear, 2016, p. 170)

This illustrative language test revealed a shortcoming of the library as developed to this point, namely, that lexical rule types could only specify features on the sole object. To enable combining valence-changing operations with argument optionality features for all operations, I modified my library and the questionnaire to allow verbal lexical rule types to specify features on the added object; the library then uses the selected position of the added object (at the front or rear of the COMPS list) to apply the constraint to the proper slot. This change also required adding limited support for second object targets to the existing features mechanism. The relevant choices file

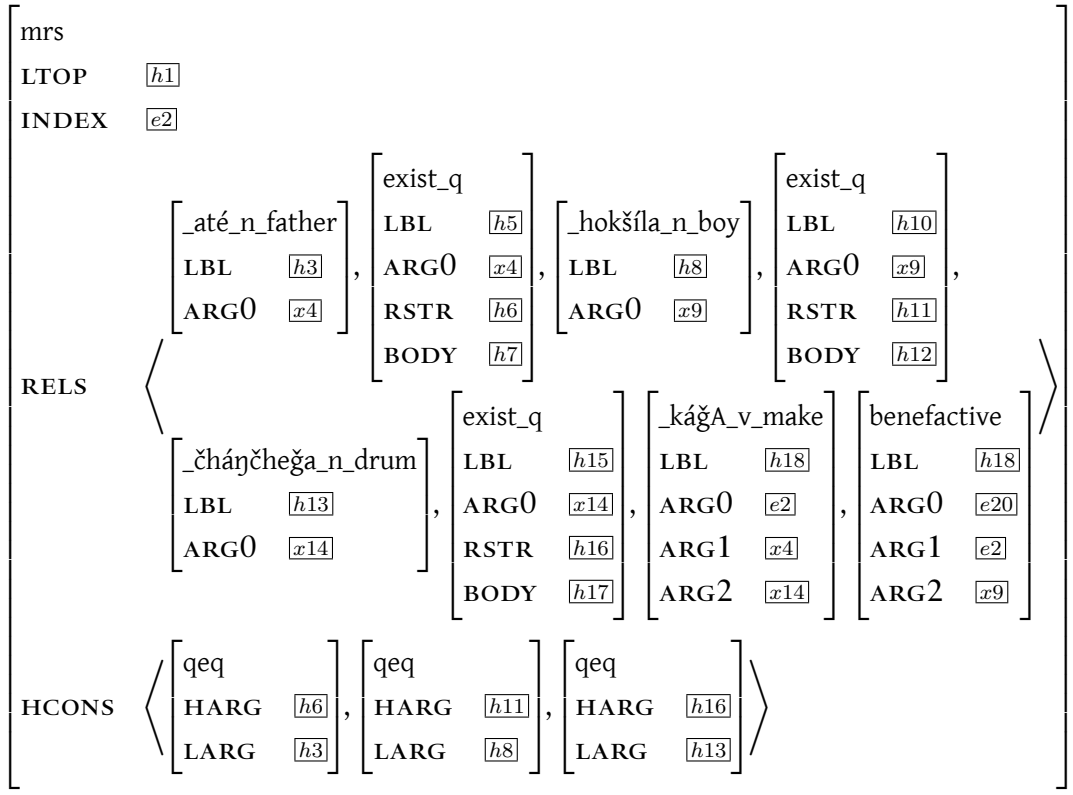
[7] Lakota does not mark for syntactic case; "dative" here, following the convention of *e.g.* Ullrich, refers simply to ditransitivity.

fragment implementing the benefactive is shown in (59),[8] and illustrates this added capability, which is activated by `feat1_head=newobj`.

(59)
```
verb-pc7_name=bene-tr-pos
verb-pc7_order=suffix
verb-pc7_inputs=verb-pc1, verb-pc4
  verb-pc7_lrt1_name=bene
    verb-pc7_lrt1_feat1_name=OPT
    verb-pc7_lrt1_feat1_value=minus
    verb-pc7_lrt1_feat1_head=newobj
    verb-pc7_lrt1_valchg1_operation=obj-add
    verb-pc7_lrt1_valchg1_inputs=trans
    verb-pc7_lrt1_valchg1_predname=benefactive_rel
    verb-pc7_lrt1_valchg1_argpos=post
    verb-pc7_lrt1_valchg1_argtype=np
    verb-pc7_lrt1_lri1_inflecting=yes
    verb-pc7_lrt1_lri1_orth=-BEN
```

As described in my analysis in Chapter 4, subject- and object-adding operations involve the addition of new elementary predications (EPs) to the semantic representation. An example of such an addition is provided in (60) below, depicting the MRS output for (58b).

[8] In the implemented test grammar, the benefactive, along with the causative, is regularized to appear as a suffix, following the approach followed in my earlier morphophonological analyzer (Curtis, 2014).

(60)


To create a suite of test sentences for Lakota, I used positive examples collected from source material, including those illustrated above, with occasional modifications by me to remove unrelated complications. I then created ungrammatical example sentences, partly based on the source examples, to test potential ways in which the expected behavior could fail. In total I created 30 examples, 16 positive and 14 negatives. The results for these tests are collected with all illustrative language test results in Table 6.2 below.

6.2.2 Japanese

Japanese [jpn], spoken by approximately 128 million native speakers (Lewis et al., 2013), also exhibits interesting valence change phenomena. Although its verbal morphology is relatively simple as compared to Lakota, unlike Lakota its operations include case frame alternations, and so it illustrates a slightly different subspace of valence-changing typology. One example is the

passive, previously described in my analysis of subject demotion (see Chapter 4). For convenience, I reproduce that example (34) here as (61):

(61) a. inu ga neko wo otta
inu ga neko wo ot-ta
dog NOM cat ACC chase-PST
'The dog chased the cat.' [jpn]

b. neko ga inu ni owareta
neko ga inu ni o-ware-ta
cat NOM dog DAT chase-PASS-PST
'The cat was chased by the dog.' [jpn] (Bender, 2013, p. 103)

For my Japanese test cases I adapted the Japanese sample choices file that is installed by default with the web customization system; specifically, I added valence-changing operations and morphemes to implement the causative and the passive, along with some additional lexical entries.

To implement the passive, I wrote a lexical rule type with two valence-changing operations: subject-demoting and object-promoting. This required some adjustments to rule generation code in my library and its interface with the morphotactics system to properly combine the operations into a single rule with the proper supertypes. The relevant choices file section is shown in (62) and shows the combination of multiple valence-changing operations in a single lexical rule type, as well as specifying case constraints on the output subject and object.

(62)
```
verb-pc2_lrt2_name=passive-tr
  verb-pc2_lrt2_feat1_name=case
  verb-pc2_lrt2_feat1_value=dat
  verb-pc2_lrt2_feat1_head=obj
  verb-pc2_lrt2_feat2_name=case
  verb-pc2_lrt2_feat2_value=nom
  verb-pc2_lrt2_feat2_head=subj
  verb-pc2_lrt2_valchg1_operation=subj-dem
  verb-pc2_lrt2_valchg1_inputs=trans
  verb-pc2_lrt2_valchg1_argpos=post
  verb-pc2_lrt2_valchg2_operation=obj-prom
  verb-pc2_lrt2_valchg2_inputs=trans
  verb-pc2_lrt2_valchg2_argpos=post
  verb-pc2_lrt2_lri1_inflecting=yes
  verb-pc2_lrt2_lri1_orth=ware
```

I also implemented the causative for Japanese, as illustrated in (63) below. This implementation worked with only one minor change to my library, namely, allowing subject-adding operations (as well as object-adding) operations to specify the HEAD type of the subject. This appears in the extract of the corresponding choices file in (64) below as `valchg1_argtype=pp`.[9]

(63) a. Suzuki ga nattou wo tabeta
Suzuki ga nattou wo tabe-ta
Suzuki NOM fermented.soybeans ACC eat-PST
'Suzuki ate natto (fermented soybeans).' [jpn] (Bender, 2013, p. 98)

b. Tanaka ga Suzuki ni nattou wo tabesaseta
Tanaka ga Suzuki ni nattou wo tabe-sase-ta
Tanaka NOM Suzuki DAT fermented.soybeans ACC eat-CAUS-PST
'Tanaka made Suzuki eat natto (fermented soybeans).' [jpn] (Bender, 2013, p. 98)

[9]This implementation of a Japanese grammar uses case-marking adpositions.

(64)
```
verb-pc3_name=caus-pos
verb-pc3_order=suffix
verb-pc3_inputs=verb
  verb-pc3_lrt1_name=caus-tr
    verb-pc3_lrt1_feat1_name=case
    verb-pc3_lrt1_feat1_value=dat
    verb-pc3_lrt1_feat1_head=newobj
    verb-pc3_lrt1_valchg1_operation=subj-add
    verb-pc3_lrt1_valchg1_inputs=trans
    verb-pc3_lrt1_valchg1_predname=cause_rel
    verb-pc3_lrt1_valchg1_argpos=post
    verb-pc3_lrt1_valchg1_argtype=pp
    verb-pc3_lrt1_lri1_inflecting=yes
    verb-pc3_lrt1_lri1_orth=sase
```

Japanese also allows the causative to be passivized, as in (65b). Interestingly, it also allows the passive to be causativized, illustrated in (65c).

(65) a. Mitiko ga Taroo ni hon wo yomaseta
Mitiko ga Taroo ni hon wo yom-sase-ta
Mitiko NOM Taroo DAT book ACC read-CAUS-PST
'Mitiko caused Taroo to read a book.' [jpn] (Heycock, 1987, p. 1)

b. Taroo ga Mitiko ni hon wo yomaserareta
Taroo ga Mitiko ni hon wo yom-sase-rare-ta
Taroo NOM Mitiko DAT book ACC read-CAUS-PASS-PST
'Taroo was made to read a book by Mitiko.' [jpn] (Heycock, 1987, p. 2)

c. Mary wa Taroo wo Ziroo ni homeraresaseta
Mary wa Taroo wo Ziroo ni home-rare-sase-ta
Mary TOP Taroo ACC Ziroo DAT praise-PASS-CAUS-PST
'Mary caused Taroo to be praised by Ziroo.' [jpn]
(Marantz, 1985, in Baker, 1985, p .639)

Implementing the combined causative and passive revealed an erroneous assumption in my original subject-demoting and object-promoting operations; namely, the assumption that these operations did not need to be parameterized to target specific positions on the COMPS list, in the same way that many of the other rule components were. Implementing this parameterization necessitated additional reworking of the functions that build the those rule components. Unlike the other rule components, however, the subject-demoting and object-promoting operations do

not alter the length of the COMPS list and so need to also ensure that any unaffected complement is maintained.

The choices file fragment in (66) that implements the passive portion of (65b) above illustrates this modification in action: the subject-demoting and object-promoting operations are specified with `inputs=ditrans` to reflect that this rule type is applied after the causative adds an argument (the output of `verb-pc3`);[10] `argpos=post` then results in the generated passive coindexing the correct arguments with their proper semantic roles.

(66)
```
verb-pc2_inputs=verb, verb-pc3
  verb-pc2_lrt1_name=passive-ditr
    verb-pc2_lrt1_feat1_name=case
    verb-pc2_lrt1_feat1_value=dat
    verb-pc2_lrt1_feat1_head=newobj
    verb-pc2_lrt1_feat2_name=case
    verb-pc2_lrt1_feat2_value=nom
    verb-pc2_lrt1_feat2_head=subj
    verb-pc2_lrt1_valchg1_operation=subj-dem
    verb-pc2_lrt1_valchg1_inputs=ditrans
    verb-pc2_lrt1_valchg1_argpos=post
    verb-pc2_lrt1_valchg2_operation=obj-prom
    verb-pc2_lrt1_valchg2_inputs=ditrans
    verb-pc2_lrt1_valchg2_argpos=post
    verb-pc2_lrt1_lri1_inflecting=yes
    verb-pc2_lrt1_lri1_orth=ware
```

The test sentences for Japanese were developed in the same manner as those for Lakota, resulting in a total of 21 sentences: 16 positive and 9 negative. These results are also included in Table 6.2.

6.2.3 Zulu

Zulu [zul] is a language in the Bantu family, spoken by approximately 27 million speakers, primarily in South Africa (Lewis et al., 2013). Zulu, like many Bantu languages, has a rich system of applicatives, which led me to select it to further expand the typological coverage of my library.

[10] Another lexical rule type in this position class implements the transitive passive, so the position class also accepts a bare verb as input.

The prototypical applicative, of course, is the benefactive, several examples of which can be found above. The Zulu benefactive is indicated by the verbal suffix *-el*, which is homophonous with the suffixes for motive, circumstance, and the locative (Buell, 2005). Examples of each are shown in (67)–(69) below.

(67) a. Ngilahla udoti
Ngi-lahl-a u-doti
1S.SBJ-dispose.of-FV 1-1.trash
'I'm taking out the trash.' [zul] (Buell, 2005, p. 189)

b. **Benefactive:**

Ngilahlela uThandi udoti
Ngi-lahl-el-a u-Thandi u-doti
1S.SBJ-dispose.of-APPL-FV 1-1.Thandi 1-1.trash
'I'm taking out the trash for Thandi.' [zul] (Buell, 2005, p. 189)

c. **Motive:**

Ngilahlela imali udoti
Ngi-lahl-el-a i-mali u-doti
1S.SBJ-dispose.of-APPL-FV 9-9.money 1-1.trash
'I'm taking out the trash for money.' [zul] (Buell, 2005, p. 189)

(68) a. Lo muntu uzofa
Lo muntu u-zo-fa
that.1 person 1.SBJ-FUT-die
'That person will die.' [zul] (Buell, 2005, p. 189)

b. **Circumstance:**

Lo muntu uzofela ukuthanda abantwana bakhe
Lo muntu u-zo-f-el-a u-ku-thanda a-bantwana ba-khe
that.1 person 1.SBJ-FUT-die-APPL-FV 15-15-love 2-2.child 2.of-1.PRON
'That person will die from loving his children (so much).' [zul]
(Buell, 2005, p. 189)

These suffixes can also license different syntactic structures, as in (69); the added argument also appear as either a PP or NP (Buell, 2005).

(69) a. Abantwana bazofunda isiBhunu
A-bantwana ba-zo-fund-a i-siBhunu
2-2.child 2.SBJ-FUT-study-FV 7-7.Afrikaans
'The children will study Afrikaans.' [zul] (Buell, 2005, pp. 189–190)

b. Abantwana bazofundela isiBhunu esikoleni
A-bantwana ba-zo-fund-el-a i-siBhunu e-sikole-ni
2-2.child 2.SBJ-FUT-study-APPL-FV 7-7.Afrikaans LOC:7-7.school-LOC
'The children will study Afrikaans at school.' [zul] (Buell, 2005, pp. 190)

c. Isikole sifundela abantwana
I-sikole si-fund-el-a a-bantwana
7-7.school 7.SBJ-study-APPL-FV 2-2.child
'The children study at school.' [zul] (Buell, 2005, p. 191)

Most of these examples were straightforward to implement using the library functions developed up to this point. The different semantic senses (*e.g.* benefactive, motive, etc.) can be simply expressed by specifying distinct predicate names, with the otherwise-identical syntactic constraints being shared by the valence-changing operation building blocks added by my library. This demonstrates the effectiveness of this approach, which both reduces the need for duplicated constraints as well as making the common syntactic structure explicit in the generated grammar.

This last example, in (69c), is especially interesting, however, as when the locative argument is an NP, it behaves as the syntactic subject while serving as the added semantic argument contributed by the applicative relation (ARG2 of the added EP).

I was able to implement this construction using my library by combining four rule components: object-adding for the added argument slot and EP, subject-demoting and object-promoting to to implement the passive aspects, and finally object-removing to remove the "missing" object of the main verb.[11] The object-promoting and subject-demoting operations needed to be expanded to be able to target the newly-ditransitive verb; although this is conceptually a minor adjustment, it necessitated removing some assumptions in my earlier code and therefore a rework of the argument position-determining elements of the rule-writing machinery.

[11] This is only necessary, of course, because I retained the underlying verb as transitive in this instance.

A further problem encountered in implementing the Zulu NP locative was that my original object-removing operation was hard-coded to only target the first complement as well as constraining or copying the other complement. These added constraints conflicted, when combined into a single lexical rule type, with other valence change operations. To resolve this, I modified the lexical rule type generation to recognize when object-removing is included with other valence-changing operations and, in those cases, to only add the matrix type *unexpressed* to the specified argument, leaving the rest of the constraints to be applied by the remaining operations.

One final limitation this construction revealed was that object-adding and object-removing operations cannot be combined in the same lexical rule type. This was easily remedied by separating them into lexical rule types in different position classes that always co-occur: the first contains the object-adding operation, while the remaining operations are applied by the second lexical rule type.[12] A simplified fragment of the choices file, shown at (70), illustrates this combined latter rule type. The object-promoting and subject-demoting operations target the added locative argument (`argpos=post`), while the object-removing operation targets the erstwhile original object (`argpos=pre`).

[12] Although this implementation was technically simple, it is admittedly awkward and unsatisfying from a linguistic perspective. A future revision of the library could improve support for multiple operations in a single lexical rule type.

(70)
```
verb-pc6_inputs=tverb
      verb-pc6_lrt1_name=applicative-pre
        verb-pc6_lrt1_valchg1_operation=obj-add
        verb-pc6_lrt1_valchg1_inputs=trans
        verb-pc6_lrt1_valchg1_predname=applicative
        verb-pc6_lrt1_valchg1_argpos=pre
        verb-pc6_lrt1_valchg1_argtype=np
          verb-pc6_lrt1_lri1_inflecting=yes
          verb-pc6_lrt1_lri1_orth=el
  verb-pc8_inputs=verb-pc6
    verb-pc8_lrt1_name=passive-rule
        verb-pc8_lrt1_valchg1_operation=subj-dem
        verb-pc8_lrt1_valchg1_inputs=ditrans
        verb-pc8_lrt1_valchg1_argpos=post
        verb-pc8_lrt1_valchg1_argtype=np
        verb-pc8_lrt1_valchg2_operation=obj-prom
        verb-pc8_lrt1_valchg2_inputs=ditrans
        verb-pc8_lrt1_valchg2_argpos=post
        verb-pc8_lrt1_valchg2_argtype=np
        verb-pc8_lrt1_valchg3_operation=obj-rem
        verb-pc8_lrt1_valchg3_inputs=ditrans
        verb-pc8_lrt1_valchg3_argpos=pre
        verb-pc8_lrt1_valchg3_argtype
```

Similarly to how I developed test sentences for the other illustrative languages, I created a test suite for Zulu consisting of 8 positive and 5 negative examples, for a total of 13 test sentences. These results are included, along with the results for the other illustrative languages, in Table 6.2.

6.2.4 Illustrative languages results

The coverage of my library against the illustrative language test suites is shown in Table 6.2. With the modifications described above, I was able to achieve full coverage, with no spurious parses. Following the illustrative language testing, I froze the library code[13] and evaluated its coverage on five held-out languages. This evaluation is described in the next section.

[13]The latest version of the library code is always included as part of the Grammar Matrix customization system, located at `http://matrix.delph-in.net`.

	examples		performance			
Language	positives	negatives	parses	coverage	overgeneration	spurious ambiguity
Lakota [lkt]	16	14	16	100%	0%	0%
Japanese [jpn]	12	9	12	100%	0%	0%
Zulu [zul]	8	5	8	100%	0%	0%
Total	36	28	36	100%	0%	0%

Table 6.2: Illustrative languages test summary and performance

6.3 Evaluation on held-out languages

In the previous sections I described the pseudolanguages and illustrative languages I used during development to shape and test my library. The pseudolanguages were abstract languages created specifically to exercise certain elements of my library, while the illustrative languages were used to ensure the library capabilities are matched to the valence change shown in actual human languages. In this section I discuss my evaluation of the library against five held-out human languages from different familial and areal groups. In contrast to the earlier languages, these languages were not studied or referenced during library development and no changes were made to the library to accommodate the phenomena identified. The full results of this evaluation are presented at the end of this section in Table 6.3 below.

6.3.1 Tsez

Tsez [ddo] is a Northeast Caucasian (Nakh-Daghestanian) language with approximately 13,000 speakers who live primarily in southwestern Dagestan, Russia (Lewis et al., 2013). It is head-final, and though it has nominally free constituent order, in practice it is essentially SOV (Comrie, 2000). A common pattern in Tsez is for transitive forms of verbs to be derived from intransitive base forms via the causative suffix *-r*, as in (71):

(71) a. meši bexus
meši b-exu-s
calf.ABS III-die-PST.WIT
'The calf died.' [ddo] (Comrie, 2000, p. 367)

b. aɬā meši bexursi
aɬ-ā meši b-exu-r-si
shepherd-ERG calf.ABS III-die-CAUS-PST.WIT
'The shepherd killed the calf.' [ddo] (Comrie, 2000, p. 367)

Transitive verbs can also be causativized. Interestingly, as shown in (72) below, causativized forms of both transitives (72a) and intransitives[14] (72b) can themselves be again casuativized.

(72) a. učitelā užiq kidbeq keč' qˁaƛirersi
učitel-ā uži-q kidb-eq keč' qˁaƛi-r-er-si
teacher-ERG boy-POSS girl-POSS song.ABS sing-CAUS-CAUS-PST.WIT
'The teacher made the boy make the girl sing a song.' [ddo] (Comrie, 2000, p. 369)

b. di žek'uq reƛ'a ritirerāčin
di žek'u-q reƛ'a r-iti-re-r-āčin
I man-POSS hand.ABS IV-touch-CAUS-CAUS-FUT.DEF.NEG
'I will not allow the man to let his hand touch (me).' [ddo] (Comrie, 2000, p. 369)

Tsez also presents an accidental construction, in which the accidental agent is marked with the possessive case: contrast (73b) with (73c). Although there is no overt verbal morphology marking of the accidental, I attempted to implement this using a non-inflecting lexical rule type.

(73) a. č'ikay yexus
č'ikay y-exu-s
glass.ABS II-break-PST.WIT
'The glass broke.' [ddo] (Comrie, 2000, p. 365)

b. užā čikay yexursi
už-ā čikay y-exu-r-si
boy-ERG glass.ABS II-break-CAUS-PST.WIT
'The boy broke the glass.' [ddo] (Comrie, 2000, p. 365)

[14] Comrie 2000 notes that *iti* "to touch" is intransitive; if the patient were expressed it would appear in the possessive case.

c. užiq č'ikay yexus
uži-q č'ikay y-exu-s
boy-POSS glass.ABS II-break-PST.WIT
'The boy accidentally broke the glass.' [ddo] (Comrie, 2000, p. 365)

The test suite I created for Tsez consists of 11 positive examples and 8 negative examples, for a total of 19 test sentences. A grammar created using only the customization system with my library was able to parse 10 of the positive examples, with no spurious parses of negative examples. Two interesting instances to implement were the double causative in (72) and the accidental construction in (73c) above. The accidental construction was straightforward to implement with a non-inflecting lexical rule that applied the subject-adding operation and specified the possessive case on the added subject.

In the case of the double causative, I needed to create an additional position class and associated lexical rule types for the second causative, as lexical rules in the Grammar Matrix cannot feed themselves. This position class also needed morphotactic rules to require the presence of the first causative position class and forbid the accidental causative lexical rule. This forbid constraint was necessary to block a single causative morpheme *-r-* from being interpreted as a second causative after the noninflected accidental causative *-Ø-r-*. The MRS produced by the double causative in (72b) is illustrated in (74).

(74)
$$\begin{bmatrix}
\text{mrs} \\
\textsc{ltop} & \boxed{h1} \\
\textsc{index} & \boxed{e2} \\
\textsc{rels} & \left\langle \begin{bmatrix} \text{pron} \\ \textsc{lbl} & \boxed{h3} \\ \textsc{arg0} & \boxed{x4} \end{bmatrix}, \begin{bmatrix} \text{exist_q} \\ \textsc{lbl} & \boxed{h5} \\ \textsc{arg0} & \boxed{x4} \\ \textsc{rstr} & \boxed{h6} \\ \textsc{body} & \boxed{h7} \end{bmatrix}, \begin{bmatrix} \text{_žek'_n_man} \\ \textsc{lbl} & \boxed{h8} \\ \textsc{arg0} & \boxed{x9} \end{bmatrix}, \begin{bmatrix} \text{exist_q} \\ \textsc{lbl} & \boxed{h10} \\ \textsc{arg0} & \boxed{x9} \\ \textsc{rstr} & \boxed{h11} \\ \textsc{body} & \boxed{h12} \end{bmatrix}, \right. \\
& \left. \begin{bmatrix} \text{_reƛ'a_n_hand} \\ \textsc{lbl} & \boxed{h13} \\ \textsc{arg0} & \boxed{x14} \end{bmatrix}, \begin{bmatrix} \text{exist_q} \\ \textsc{lbl} & \boxed{h15} \\ \textsc{arg0} & \boxed{x14} \\ \textsc{rstr} & \boxed{h16} \\ \textsc{body} & \boxed{h17} \end{bmatrix}, \begin{bmatrix} \text{_iti_v_hit} \\ \textsc{lbl} & \boxed{h18} \\ \textsc{arg0} & \boxed{e19} \\ \textsc{arg1} & \boxed{x14} \end{bmatrix}, \right\rangle \\
& \begin{bmatrix} \text{casuative} \\ \textsc{lbl} & \boxed{h20} \\ \textsc{arg0} & \boxed{e21} \\ \textsc{arg1} & \boxed{x9} \\ \textsc{arg2} & \boxed{x14} \\ \textsc{arg3} & \boxed{h22} \end{bmatrix}, \begin{bmatrix} \text{causative} \\ \textsc{lbl} & \boxed{h23} \\ \textsc{arg0} & \boxed{e2} \\ \textsc{arg1} & \boxed{x4} \\ \textsc{arg2} & \boxed{x9} \\ \textsc{arg3} & \boxed{h24} \end{bmatrix}, \begin{bmatrix} \text{neg} \\ \textsc{lbl} & \boxed{h25} \\ \textsc{arg0} & \boxed{e27} \\ \textsc{arg1} & \boxed{h26} \end{bmatrix} \\
\textsc{hcons} & \left\langle \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h6} \\ \textsc{larg} & \boxed{h3} \end{bmatrix}, \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h11} \\ \textsc{larg} & \boxed{h8} \end{bmatrix}, \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h16} \\ \textsc{larg} & \boxed{h13} \end{bmatrix}, \right. \\
& \left. \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h22} \\ \textsc{larg} & \boxed{h18} \end{bmatrix}, \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h24} \\ \textsc{larg} & \boxed{h20} \end{bmatrix}, \begin{bmatrix} \text{qeq} \\ \textsc{harg} & \boxed{h26} \\ \textsc{larg} & \boxed{h23} \end{bmatrix} \right\rangle
\end{bmatrix}$$

The one sentence that failed to parse was the causativized causative of a transitive, shown above in (72a). This was caused by the requirement of my library for valence-changing lexical rule types to specify their inputs as either strictly intransitive or strictly transitive. In this instance, since the input verb is transitive, the output of the first causative lexical rule is ditransitive, which is not currently supported by my library as an input for the second causative. The

results of this evaluation of the library using Tsez are summarized with the other test languages in Table 6.3 below.

6.3.2 *West Greenlandic*

West Greenlandic [kal], or Kalaallisut, is a language in the Eskimo-Aleut family with approximately 44,000 speakers, primarily living in Greenland, where it is the statutory national language (Lewis et al., 2013). It is highly polysynthetic, with a nominally free but pragmatically neutral SOV word order (Fortescue, 1993).

West Greenlandic has a system of several antipassive affixes which have a detransitivizing effect as well as requiring instrumental case on the object (Bittner, 1987). A typical example using the *-si* antipassive morpheme is shown in (75).[15]

(75) a. Jaakup ujarak tiguaa
Jaaku-p ujarak tigu-a-a
Jacob-ERG stone.ABS take-TR.IND-3SG.ERG>3SG.ABS
'Jacob took stone' [kal] (Bittner, 1987, p. 194)

b. Jaaku ujaqqamik tigusivuq
Jaaku ujarak-mik tigu-si-vu-q
Jacob.ABS stone-INS take-ANTIP-INTR.IND-3SG.ABS
'Jacob took stone' [kal] (Bittner, 1987, p. 194)

The antipassive affix *-si* can also completely remove the object, as in (76a). Another antipassive affix, *-llir* "just beginning to", can be considered to indicate a progressive aspect, shown in (b). In addition, there also exists a null-affix form of the antipassive, as in (76c).

(76) a. Kaali toqutsivoq
Kaali toqu-si-pu-q
Kaali.ABS kill-ANTIP-INTR.IND-3SG.ABS
'Kaali is a killer.' [kal] (Schmidt, 2004, p. 390)

[15] Bittner (1987) notes that the definiteness status of in West Greenlandic is controversial and omits articles from her translations; I defer to her translations to maintain consistency with the cited examples.

b. atuakkamik taassuminnga aturlirpuq
atuagak-mik taa-ssuminnga atur-llir-pu-q
book-INST this-SG.INST use-ANTIP.PROG-INTR.IND-3SG.A
'He's just now asking whether he can use this book.' [kal] (Bittner, 1987, p. 201)

c. Jaaku illumik taassuminnga sanavuq
Jaaku illu-mik taa-ssuminnga sana-Ø-pu-q
Jacob house-INST this-SG.INST build-ANTIP-INTR.IND-3SG.A
'Jacob was/is building this house.' [kal] (Bittner, 1987, p. 202)

These examples of the antipassive exposed a gap in the coverage of my library in its current status. Specifically, the existing mechanisms do not allow case change on an argument that does not otherwise change its syntactic role. This gap accounted for all of the grammatical examples in my test suite that failed to parse (4 of 15).

West Greenlandic also has a causative, with three attested variations (Underhill, 1980). These variations provide different case frames and constituent order while exhibiting the same semantic effect. When the erstwhile subject is in the absolutive, it can only be in the most-oblique position of the complements; in the allative case it can be either more- or less-oblique than the object, which remains the semantic patient. These are contrasted in (77b–c) and (77d–e) below.

(77) a. Annap immusuaq nerivaa
Anna-p immusuaq neri-pa-a
Anna-ERG cheese.ABS eat-TR.IND-3SG.E>3SG.A
'Anna ate the cheese.' [kal]

b. Piitap Anna immusuaq neritippaa
Piita-p Anna immusuaq neri-tip-pa-a
Peter-ERG Anna.ABS cheese.ABS eat-CAUS-TR.IND-3SG.E>3SG.A
'Peter made Anna eat the cheese.' [kal]

c. *Piitap immusuaq Anna neritippaa
Piita-p immusuaq Anna neri-tip-pa-a
Peter-ERG cheese.ABS Anna.ABS eat-CAUS-TR.IND-3SG.E>3SG.A
'Peter made Anna eat the cheese.' [kal]

d. Piitap Annamut immusuaq neritippaa
Piita-p Anna-mut immusuaq neri-tip-pa-a
Peter-ERG Anna-ALL cheese.ABS eat-CAUS-TR.IND-3SG.E>3SG.A
'Peter made Anna eat the cheese.' [kal]

e. Piitap immusuaq Annamut neritippaa
Piita-p immusuaq Anna-mut neri-tip-pa-a
Peter-ERG cheese.ABS Anna-ALL eat-CAUS-TR.IND-3SG.E>3SG.A
'Peter made Anna eat the cheese.' [kal] (Underhill, 1980, p. 474)

These causatives were straightforward to implement using my library, with one minor quirk. Specifically, the limitation on the position of the erstwhile subject in the absolutive form in (77b–c) required the addition of an animacy constraint to block the negative example in (77c).

Finally, West Greenlandic passive can be expressed in several variations, as shown in (78). These forms were straightforward to implement using the existing subject-demoting and -removing and object-promoting operations available in my library.

(78) a. timmiaq qimmimik nerisaavoq
timmiaq qimmi-mik neri-saa-vo-q
bird.ABS dog-INST eat-PASS-INTR.IND-3SG.A
'The bird was eaten by the dog.' [kal] (Underhill, 1980, p. 475)

b. timmiaq qimmimik neritippoq
timmiaq qimmi-mik neri-tip-po-q
bird.ABS dog-INST eat-PASS-INTR.IND-3SG.A
'The bird was eaten by the dog.' [kal] (Underhill, 1980, p. 475)

c. timmiaq nerineqarpoq
timmiaq neri-neqar-po-q
bird.ABS eat-PASS-INTR.IND-3SG.A
'The bird was eaten.' [kal] (Underhill, 1980, p. 476)

In total, my test suite for West Greenlandic included 15 positive examples, of which I was able to parse 11, and 14 negative examples, with one unexpected parse. These results are included in the summary in Table 6.3.

6.3.3 Awa Pit (Cuaiquer)

Awa Pit [kwi] is a language in the Barbacoan family, spoken by approximately 12,000 speakers (Lewis et al., 2013) on the western slopes of the Andes mountains in southwestern Colombia and northwestern Ecuador (Curnow, 1997). It is SOV, with a nominative-accusative case system.

The valence-changing constructions of Awa Pit are limited to the causative and syntactically-identical forms, such as the assistive or "auxiliative" (Curnow, 1997), and were generally straightforward to implement using the same mechanisms previously described. An interesting case, however, is presented by the combination of the causative and desiderative, illustrated in (79):

(79) a. nana Juanta wakata payninshimtus
na-na Juan-ta wakata pay-nin-shi-mtu-s
1SG.NOM-TOP Juan-ACC cattle buy-CAUS-DES-IMPF-LOCUT
'I want to sell cattle to Juan (cause Juan to buy cattle).' [kwi]
(Curnow, 1997, p. 158)

b. nana Juanta wakata payshininmtus
na-na Juan-ta wakata pay-shi-nin-mtu-s
1SG.NOM-TOP Juan-ACC cattle buy-DES-CAUS-IMPF-LOCUT
'I am making Juan want to buy cattle.' [kwi] (Curnow, 1997, p. 158)

Notionally, this should be implementable by considering the desiderative as a scopal relation like the causative. These specific attested examples are not implementable for the same reason as the double causative in Section 6.3.1 above, namely, that my library cannot accept a ditransitive (causative of a transitive) as the input to a causative operation. However, a more fundamental limitation is that the desiderative does not actually change valence and so is out of scope for my library. Nevertheless, because it could leverage the scopal relation mechanism that my library uses for the causative, support for constructs like the desiderative are likely relatively simple to implement, and thus further validate the usefulness of my building-block approach.

In total, my Awa Pit test suite includes 14 sentences, with 7 positive and 7 negative examples. As discussed in this section, I was unable to customize a grammar for the causative-desiderative combination, which accounted for all of the parse failures (2 of 7). In addition, there were no unexpected parses or spurious ambiguity. These results are included in Table 6.3 with results from the other held-out test languages.

6.3.4 Rawang

Rawang[16] [raw] is a Sino-Tibetan language spoken in the border region of China and Myanmar (LaPolla, 2000) with approximately 60,000 speakers (Lewis et al., 2013). It has a rich variety of valence-changing morphology, including both valence-decreasing and valence-increasing affixes.

LaPolla (2000) presents two valence-decreasing affixes in Rawang, the intransitivizing prefix *v-*, glossed as PREF, and the reflexive/middle marker *-shì*, glossed as R/M. LaPolla also argues that the intransitivizing prefix can function as a reciprocal when the sole remaining argument is plural and animate, as shown in (80).

(80) a. àngmaq vshvtnē
àngmaq vshvtnē
3PL PREF-hit/kill-NPST
'They are arguing/fighting' [raw]

b. àngmaq vyv̀ngkēē
àngmaq v-yv̀ng-kē-ē
3PL PREF-see-RECIP-NPST
'They are looking at each other.' [raw] (LaPolla, 2000, p. 288)

In the first example (80a), however, it is not obviously the case that the argument is syntactically reciprocal; in the second example (80b) the reciprocal is at least partially expressed by an additional affix. Accordingly, I treat these examples as out of scope for my library at present.

The reflexive/middle marker, illustrated in (81), has several functions in addition to removing a valence slot. It can mark a true reflexive or, with an added NP, a possessive reflexive; it can also be used to construct middles, as in (81c); Rawang does not have a true passive.

(81) a. àng (nø̄ àng) vdipshìē
àng nø̄ àng vdip-shì-ē
3SG TOP 3SG hit-R/M-NPST
'He is hitting himself.' [raw] (LaPolla, 2000, p. 289)

[16]The language has also commonly been referred to as Dulong, especially by speakers in China. The trend in the speaker community is to refer to it as Rawang (LaPolla, 2000), and that is the term I use here.

b. àng mv́r zv́lshìē
àng mv́r zv́l-shì-ē
3SG face wash-R/M-NPST
'He is washing his face.' [raw] (LaPolla, 2000, p. 291)

c. àng vhø̄shìē
àng vhø̄-shì-ē
3SG laugh-smile-R/M-NPST
'He is laughing.' [raw] (LaPolla, 2000, p. 290)

In the case of the reflexive in (81a), my library in its current state does not have a mechanism for specifying the same NP as the semantic agent (ARG1) and patient (ARG2) of the verb, although it appears that it would be a relatively straightforward addition. As for the possessive reflexive, I analyze it as principally about the possessive, with no actual syntactic effect on valence and so not in scope. The middle illustrated in (81c) can, however, be analyzed as a simple valence-reducing process.

The reflexive/middle affix *-shì* can also act in a way more akin to a benefactive, as in (82), with the desired MRS fragment in (82c).

(82) a. à:ngí shvmǿ sha:tnòē
àng-í shvmǿ shvt-ò-ē
3SG-A mosquito kill-3+TR.NPST-NPST
'He is killing a mosquito.' [raw] (LaPolla, 2000, p. 292)

b. àng shvmǿ shvtshìē
àng shvmǿ shvt-shì-ē
3SG mosquito kill-R/M-NPST
'He is killing a mosquito (on him).' [raw] (LaPolla, 2000, p. 292)

c.
$$\begin{bmatrix} \text{mrs} \\ \text{RELS} \left\langle \begin{bmatrix} \text{pron} \\ \text{LBL} & h1 \\ \text{ARG0} & x1 \end{bmatrix}, \begin{bmatrix} \text{\_ shvmǿ\_n\_mosquito} \\ \text{LBL} & h2 \\ \text{ARG0} & x2 \end{bmatrix}, \begin{bmatrix} \text{\_shvt\_v\_kill} \\ \text{LBL} & h3 \\ \text{ARG0} & e1 \\ \text{ARG1} & x1 \\ \text{ARG2} & x2 \end{bmatrix}, \begin{bmatrix} \text{benefactive} \\ \text{LBL} & h4 \\ \text{ARG0} & e2 \\ \text{ARG1} & e1 \\ \text{ARG2} & x1 \end{bmatrix} \right\rangle \end{bmatrix}$$

Although most of the building blocks necessary to implement this construction do exist in my library, as described above for reflexives, the key function of filling a valence slot while coindexing with an existing argument does not yet exist. Therefore, I was unable to implement these examples.

Rawang does have a true argument-adding benefactive as well, marked by the suffix *-a*, which can apply to both intransitive and transitives. When applied to transitives, the new argument is also marked with either a benefactive postposition *dvpvt* or locative postposition *svng*. The benefactive is shown in (83):

(83) ngàí àng svng shǿng rǿngāngòē
ngà-í àng svng shǿng rí-ng-ā-ng-ò-ē
1SG-AGT 3SG LOC wood carry-1SG-BEN-1SG-3+TR.NPST-NPST
'I'm carrying wood for him.' [raw] (LaPolla, 2000, p.305)

Both an affixing and periphrastic causative are possible in Rawang; as my focus in this work is on morphological valence change, I do not address the periphrastic causative. The causative prefix *dv-* is shown in (84).

(84) à:ngí Vpūng svng laqtūn dvgwāòē
àng-í Vpūng svng laqtūn dv-gw-ā-ò-ē
3SG-AGT Vpung LOC clothing CAUS-put.on-3+TR.NPST-NPST
'He made (or helped) Vpung put his clothes on.' [raw] (LaPolla, 2000, p.298)

This causative form is structurally the same as previously-addressed causatives and was simple to implement using my existing library mechanisms. There is also an interaction between the causative and the reflexive/middle marker. This combination is illustrated below in (85):

(85) a. à:ngí laqtūn dvshúòē
àng-í laqtūn dv-shū-ò-ē
3SG-INST clothing CAUS-be.dry-3+TR.NPST-NPST
'He is drying clothes.' [raw] (LaPolla, 2000, p.300)

b. àng laqtūn dvshūshìē
àng laqtūn dv-shū-shì-ē
3SG clothing CAUS-be.dry-R/M-NPST
'He is drying his clothes.' [raw] (LaPolla, 2000, p.300)

For the same reasons as described earlier, the absence of a mechanism for specifying coindexation, this construction is not implementable directly using my library in this form.

In total I created 17 test sentences in Rawang, consisting of 11 positive and 6 negative examples. Due primarily to the missing reflexive/coindexing operation, 5 positive examples either failed to generate correct parses or failed to parse. These results are also summarized in Table 6.3 in this section.

6.3.5 Javanese

Javanese [jav] is an Austronesian language with approximately 84 million speakers who live primarily in the central and eastern regions of Java, the most populous island in Indonesia (Lewis et al., 2013). Javanese exhibits SVO word order and has the symmetrical agentive-objective voice alternation typical of the Indonesian subgroup of Austronesian languages (Arka, 2002).[17] For the purposes of this work, this alternation can be treated as a form of the passive, as illustrated in (86):

(86) a. aku masak jajan kanggó Karolina
1SG AV.cook cake for Karolina
'I baked a cake for Karolina' [jav] (Arka, 2002, p. 168)

b. jajan dimasak (kanggó Karolina)
jajan di-masak (kanggó Karolina)
cake OV-cook (for Karolina)
'A cake was baked (for Karolina).' [jav] (Arka, 2002, p. 169)

The applicative appears in Javanese as both the locative suffix *-i* and the benefactive suffix *-até*, illustrated in (87b) and (87d), respectively. Both applicatives can also be passivized, as in (87c) and (87e).

(87) a. pelem ceblòk menyang gentèng ómahku
pelem ceblòk menyang gentèng ómah-ku
mango fall towards roof house-1SG.POSS
'A mango fell on the roof of my house.' [jav] (Arka, 2002, p. 168)

[17] The agentive voice (AV) and objective voice (OV) both appear in inflected and bare verb forms; the agentive- and objective-marking prefixes exhibit complex morphophonology which I have regularized in my test suite.

b. pelem nyeblòki gentèng ómahku
pelem nyeblòk-i gentèng ómah-ku
mango AV.fall-LOC roof house-1SG.POSS
'A mango fell on the roof of my house.' [jav] (Arka, 2002, p. 168)

c. gentèng ómahku diceblòki pelem
gentèng ómahku di-ceblòk-i pelem
roof house-1SG.POSS OV-fall-LOG mango
'The roof of my house had a mango foll on it.' [jav] (Arka, 2002, p. 168)

d. aku masakaké Karolina jajan
aku masak-aké Karolina jajan
1SG AV.cook-BEN Karolina cake
'I baked a cake for Karolina.' [jav] (Arka, 2002, p. 168)

e. Karolina dimasakaké jajan
Karolina di-masak-alé jajan
Karolina OV-cook-BEN cake
'Karolina was baked a cake.' [jav] (Arka, 2002, p. 168)

There are also homophonous suffixes *-i* and *-até* that function as causatives, for example as shown in (88). Arka (2002) notes that both the applicative and causative have "morphosemantic" functions and restrictions; I focus here on the morphosyntax of these affixes.

(88) a. kucing mangan iwak
cat AV.eat fish
'The cat ate fish.' [jav] (Arka, 2002, p. 169)

b. aku mangani kucing iwak
aku mangan-i kucing iwak
1SG AV.eat-CAUS cat fish
'I fed the cat fish.' [jav] (Arka, 2002, p. 168)

One notable construction is the indirect causative with *-até* when applied to transitive verbs, as illustrated in (89). In this usage, the erstwhile object is moved into an oblique position:

(89) aku manganaké iwak menyang kucing
aku mangan-aké iwak menyang kucing
1SG AV.eat-CAUS fish towards cat
'I fed the fish to the cat.' [jav] (Arka, 2002, p. 170)

	examples		performance			
Language	positives	negatives	parses	coverage	overgeneration	spurious ambiguity
Tsez [ddo]	11	8	10	91%	0%	0%
West Greenlandic [kal]	15	14	12	73%	0%	0%
Awa Pit [kwi]	7	7	5	71%	0%	0%
Rawang [raw]	11	6	6	55%	0%	0%
Javanese [jav]	13	8	12	92%	13%	0%
Total	57	43	45	79%	2%	0%

Table 6.3: Test languages test summary and performance

This last phenomenon was the only example I was not able to implement using my library and the customization system; while I created a mechanism for specifying the HEAD of the added argument, I did not add the ability to change the head of an already-existing argument. In my test suite for Javanese, I created 13 positive examples and 8 negative examples, for a total of 21 test sentences; this gap resulted in one positive example failing to parse and one negative example being incorrectly parsed. These results are summarized in Table 6.3.

6.4 Summary

In this chapter I have described how I developed and tested my library additions using synthetic psuedolanguages to exercise specific operations, as well as three illustrative natural languages to refine and expand the range of implementable phenomena based on real-world examples. I then showed the coverage my additions were able to reach on five held-out test languages, whose characteristics did not influence the design or development of the library. With an overall coverage of 79% across all five languages and an aggregate overgeneration rate of only 2%, these tests indicate that my library is indeed able to cover a substantial range of the cross-linguistic typological variation in valence-changing constructions. In addition, for Rawang [raw], the language with the poorest coverage (55%), most of the failures were due to the lack of a single configuration option that appears to be relatively simple to add to the library in future development. The sole

example of overgeneration, from Javanese [jav], was likewise due to the inability of the current library to apply one of the rule components to already-existing arguments, which would similarly be a simple extension to the existing code. Overall, these results appear to validate my hypothesis that a modular, "building-block" based approach is an effective way to provide significant typological coverage of valence change.

Chapter 7

CONCLUSION AND FUTURE WORK

In this thesis I have described my approach to developing and evaluating a library for valence-changing verbal morphology to the Grammar Matrix customization system. In so doing I set out to test two hypotheses. First, I hypothesized that a library such as this one, consisting of a subset of stored analyses of valence changing operations could support a meaningfully broad segment of the variation seen in the world's languages. Second, I also posited that a modular, "building-block" approach to decomposing valence change into isolated elements and reassembling them to implement the necessary operations was an effective implementation method with benefits in both library and grammar development.

My results showed that both these hypotheses are well-supported. As to the first, my aggregate parse coverage of 79% across languages from unrelated linguistic and areal groups showed that this relatively small set of operations was indeed broadly effective at implementing valence change in many different configurations. Regarding the second hypothesis, my modular approach was effective both in simplifying the work at the library development level (primary through reduction in repetitive code) and in simplifying the customized grammar output such that common elements are shared instead of duplicated across rules.

The development and evaluation of this library also highlighted several avenues for improvement and potential future work. For example, the coverage results were dominated by poor performance on Rawang [raw] due to a configuration option that was not implemented. A relatively simple addition would significantly increase the flexibility and coverage of the library. More substantial modifications and extensions may also be of interest; notably, I defined this library as limited to verbal morphology and did not address periphrastic constructions. Although it is not immediately obvious to what extent periphrastic valence change can be implemented using

the same machinery, the elements developed here may possibly provide a useful starting point from which to begin investigating such an extension.

REFERENCES

Agrawal, H., Horgan, J. R., Krauser, E. W., & London, S. A. (1993). Incremental regression testing. In *Proceedings of the 1993 Conference on Software Maintenance (CSM-93)* (pp. 348–357).

Arka, I. W. (2002). Voice systems in Austronesian languages of Nusantara: Typology, symmetricality and undergoer orientation. *Linguistik Indonesia*, *21*(1), 113–139.

Baker, M. C. (1985). *Incorporation: A Theory of Grammatical Function Changing* (Unpublished doctoral dissertation). Massachussetts Institute of Technology.

Beck, K. (2002). *Test Driven Development: By Example.* Addison-Wesley.

Bender, E. M. (2008). Grammar engineering for linguistic hypothesis testing. In N. Gaylord, A. Palmer, & E. Ponvert (Eds.), *Proceedings of the Texas Linguistics Society X Conference: Computational linguistics for less-studied languages* (pp. 16–36).

Bender, E. M. (2010). Reweaving a grammar for Wambaya. *Linguistic Issues in Language Technology*, *3*(3), 1–34.

Bender, E. M. (2013). *Linguistic fundamentals for natural language processing: 100 essentials from morphology and syntax.* Morgan & Claypool Publishers.

Bender, E. M. (2015). *(Un-)orthodoxy in use of HOOK features, semantic algebra compliance.* MRS representation issues [discussion], presented at the Eleventh DELPH-IN Summit. Retrieved from `http://moin.delph-in.net/SingaporeSemanticAlgebraCompliance`

Bender, E. M., Drellishak, S., Fokkens, A., Poulson, L., & Saleem, S. (2010). Grammar customization. *Research on Language & Computation*, *8*(1), 23–72.

Bender, E. M., Flickinger, D., & Oepen, S. (2002). The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-linguistically Consistent Broad-coverage Precision Grammars. In J. Carroll, N. Oostdijk, & R. Sutcliffe (Eds.), *Proceedings of the*

Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics (pp. 8–14). Taipei, Taiwan.

Bender, E. M., & Good, J. (2005). Implementation for discovery: A bipartite lexicon to support morphological and syntactic analysis. In *Proceedings from the annual meeting of the Chicago Linguistic Society* (Vol. 41, pp. 1–16).

Bender, E. M., & Langendoen, D. T. (2010). Computational linguistics in support of linguistic theory. *Linguistic Issues in Language Technology*, *3*(1), 1–31.

Bender, E. M., Poulson, L., Drellishak, S., & Evans, C. (2007, June). Validation and regression testing for a cross-linguistic grammar resource. In *ACL 2007 Workshop on deep linguistic processing* (pp. 136–143). Prague, Czech Republic: Association for Computational Linguistics.

Bittner, M. (1987). On the semantics of the Greenlandic antipassive and related constructions. *International Journal of American Linguistics*, *53*(2), 194–231.

Bouma, G., Malouf, R., & Sag, I. A. (2001). Satisfying constraints on extraction and adjunction. *Natural Language & Linguistic Theory*, *19*(1), 1-65.

Buell, L. C. (2005). *Issues in Zulu Verbal Morphosyntax* (Unpublished doctoral dissertation). University of California, Los Angeles.

Bybee, J. L. (1985). *Morphology: A study of the relation between meaning and form.* Amsterdam: John Benjamins Publishing.

Carpenter, B. (1992). *The logic of typed feature structures.* Cambridge, U.K.: Cambridge University Press.

Chung, S. (1976). An object-creating rule in Bahasa Indonesia. *Linguistic Inquiry*, *7*, 41–87.

Cole, P. (1982). *Imbabura Quechua.* London etc.: Croom Helm.

Cole, P., & Sridhar, S. N. (1977). Clause union and relational grammar: Evidence from Hebrew and Kannada. *Linguistic Inquiry*, *8*(4), 700-713.

Comrie, B. (2000). Valency-changing derivations in Tsez. In R. M. W. Dixon & A. Y. Aikhenvald (Eds.), *Changing valency: case studies in transitivity.* Cambridge: Cambridge University Press.

Comrie, B., & Nedjalkov, V. P. (1988). *Typology of resultative constructions: Translated from the original Russian edition (1983)* (Vol. 12). John Benjamins Publishing.

Copestake, A. (2002a). Definitions of typed feature structures. In S. Oepen, D. Flickinger, J. Tsujii, & H. Uszkoreit (Eds.), *Collaborative language engineering* (pp. 227–230). Stanford, CA: CSLI Publications.

Copestake, A. (2002b). *Implementing Typed Feature Structure Grammars.* CSLI Publications.

Copestake, A., Flickinger, D., Pollard, C., & Sag, I. A. (2005). Minimal recursion semantics: An introduction. *Research on Language and Computation*, *3*(2), 281–332.

Copestake, A., Lascarides, A., & Bender, E. M. (2016). *Algebra replacement.* Revising the algebra [discussion], presented at the Twelfth DELPH-IN Summit. Retrieved from `http://moin.delph-in.net/StanfordAlgebraAdditions`

Copestake, A., Lascarides, A., & Flickinger, D. (2001). An algebra for semantic construction in constraint-based grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics* (pp. 140–147). Stroudsburg, PA, USA: Association for Computational Linguistics.

Crowgey, J. (2012). *The syntactic exponence of sentential negation: a model for the LinGO Grammar Matrix* (Unpublished master's thesis). University of Washington.

Curnow, T. J. (1997). *A grammar of Awa Pit (Cuaiquer): An indigenous language of south-western Colombia* (Unpublished doctoral dissertation). The Australian National University.

Curtis, C. (2014). A finite-state morphological analyzer for a Lakota HPSG grammar. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014).* European Language Resources Association (ELRA).

Dayley, J. P. (1989). *Tümpisa (Panamint) Shoshone grammar* (Vol. 115). Univ of California Press.

Dixon, R. M. W. (1979). Ergativity. *Language*, *55*(1), 59-138.

Dixon, R. M. W., & Aikhenvald, A. Y. (1997). A typology of argument-determined constructions. In J. Bybee, J. Haiman, & S. A. Thompson (Eds.), *Essays on language function and language type* (pp. 71–113). Amsterdam: John Benjamins.

Dixon, R. M. W., & Aikhenvald, A. Y. (2000). *Changing valency.* Cambridge University Press.

Drellishak, S. (2009). *Widespread but not universal: Improving the typological coverage of the Grammar Matrix* (Unpublished doctoral dissertation). University of Washington.

England, N. C. (1983). *A grammar of Mam, a Mayan language*. Austin: University of Texas Press.

Fokkens, A. S. (2010). *Documentation for the Grammar Matrix word order library* (Tech. Rep.). Saarbrücken: Saarland University.

Fortescue, M. (1993). Eskimo word order variation and its contact-induced perturbation. *Journal of Linguistics*, *29*(2), 267–89.

Gerdts, D. B., & Kiyosawa, K. (2005). Halkomelem psych applicatives. *Studies in Language*, *29*, 329–362.

Goodman, M. W. (2013). Generation of machine-readable morphological rules from human-readable input. *University of Washington Working Papers in Linguistics*, *30*.

Guillaume, A. (2014). The Interaction of Reduplication with Word Classes and Transitivity in Cavineña. In G. G. Gomez & H. van der Voort (Eds.), *Reduplication in Indigenous Languages of South America* (pp. 313–342). Leiden: Brill.

Harris, A. C. (1981). *Georgian syntax: A study in relational grammar*. Cambridge: Cambridge University Press.

Haspelmath, M., & Müller-Bardey, T. (2004). Valence change. *Morphology: A handbook on inflection and word formation*, *2*, 1130–1145.

Heycock, C. (1987). *The structure of the Japanese causative* (Tech. Rep. No. MS-CIS-87-55). Department of Computer and Information Science, University of Pennsylvania.

Károly, S. (1982). Intransitive-transitive derivational suffixes in Hungarian. In F. Kiefer (Ed.), *Hungarian linguistics* (Vol. 4, pp. 185–243).

LaPolla, R. J. (2000). Valency-changing derivations in Dulong/Rawang. In R. M. W. Dixon & A. Y. Aikhenvald (Eds.), *Changing valency: case studies in transitivity* (pp. 282–311). Cambridge: Cambridge University Press.

Lewis, M. P., Simons, G. F., & Fenning, C. D. (Eds.). (2013). Ethnologue: Languages of the world, seventeenth edition. *Dallas, Tex.: SIL International. Online version: http://www. ethnologue. com.*

Manning, C. D., & Sag, I. A. (1998). Argument structure, valence, and binding. *Nordic Journal of Linguistics*, *21*(2), 107–144.

Marantz, A. P. (1985). *The Nondistinctiveness of Derivational and Inflectional Morphology.* (Unpublished paper, Harvard University, Cambridge, and University of North Carolina, Chapel Hill)

Moseley, C. (Ed.). (2010). *Atlas of the world's languages in danger* (3rd edn. ed.). Paris: UNESCO Publishing. Retrieved from `http://www.unesco.org/culture/en/endangeredlanguages/atlas`

Müller, S. (2017). *Phrasal constructions, derivational morphology, constituent structure and (cross-linguistic) generalizations: A discussion of template-based phrasal LFG approaches and a lexical HPSG alternative.* Preprint at `https://hpsg.hu-berlin.de/~stefan/Pub/phrasal-lfg.html`. (Retrieved: 2017-04-02)

Nakiboglu-Demiralp, M. (2001). The referential properties of the implicit arguments of impersonal passive constructions. In E. E. Taylan (Ed.), *The verb in Turkish* (Vol. 44, p. 129). Amsterdam: John Benjamins.

Oepen, S., & Flickinger, D. (1998). Towards systematic grammar profiling. Test suite technology 10 years after. *Computer Speech & Language*, *12*(4), 411–435.

O'Hara, K. (2008). *A morphotactic infrastructure for a grammar customization system* (Unpublished master's thesis). University of Washington.

Polinsky, M., & Kozinsky, I. (1992). Ditransitive constructions in Kinyarwanda: coding conflict or syntactic doubling? In *Papers from the 28th Annual Meeting of the Chicago Linguistic Society* (pp. 426–442).

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

Pustet, R., & Rood, D. S. (2008). Argument dereferentialization in Lakota. In M. Donohue & S. Wichman (Eds.), *The Typology of Semantic Alignment*. Oxford University Press.

Sag, I. A., Wasow, T., & Bender, E. M. (2003). *Synactic theory: A formal introduction* (2nd ed. ed.). Stanford, CA: CSLI.

Saleem, S. (2010). *Argument optionality: A new library for the Grammar Matrix customization system* (Unpublished master's thesis). University of Washington.

Saleem, S., & Bender, E. M. (2010). Argument optionality in the LinGO Grammar Matrix. In *Proceedings of the 23rd international conference on computational linguistics: Posters* (pp. 1068–1076). Stroudsburg, PA, USA: Association for Computational Linguistics.

Schaub, W. (1982). *Babungo*. London etc.: Croom Helm.

Schmidt, B. (2004). West Greenlandic antipassive. *Nordlyd 31: Proceedings of the 19th Scandinavian Conference of Linguistics*, *31*(2), 385–399.

Shibatani, M. (1990). *The languages of Japan*. Cambridge University Press.

Song, S. (2014). *A grammar library for information structure* (Unpublished doctoral dissertation). University of Washington.

Tesnière, L. (1959). *Eléments de syntaxe structurale*. Paris: Librairie C. Klincksieck.

Ullrich, J. (2011). *New Lakota dictionary*. Bloomington, IN: Lakota Language Consortium.

Ullrich, J., & Black Bear, B., Jr. (2016). *Lakota Grammar Handbook*. Bloomington, IN: Lakota Language Consortium.

Underhill, R. (1980). Case relations in modern Greenlandic. *Proceedings of the Sixth Annual Meeting of the Berkeley Linguistics Society*, 467–478.

Van Valin, R. D. (1977). *Aspects of Lakhota syntax: A study of Lakhota (Teton Dakota) syntax and its implications for universal grammar* (Unpublished doctoral dissertation). University of California.

Washio, R. (1995). *Interpreting voice. a case study in lexical semantics.* Tokyo: Kaitakusha.

Williams, L., Maximilien, E. M., & Vouk, M. (2003). Test-driven development as a defect-reduction practice. In *14th International Symposium on Software Reliability Engineering (ISSRE 2003)* (pp. 34–45).

Wunderlich, D. (2015). Valency-changing word-formation. In P. O. Müller, I. Ohnheiser, S. Olsen, & F. Rainer (Eds.), *Word-formation* (Vol. 3, pp. 1424–1466). Berlin/Boston: De Gruyter Mouton.

Appendix A

INDEX OF LANGUAGES REFERENCED

Language	Family	ISO 639-3	Pages
Ainu	Ainu	ain	15
Awa Pit	Barbacoan	kwi	77–78
English	Indo-European (West Germanic)	eng	20, 28
Georgian	Kartvelian	kat	17, 33
German	Indo-European (West Germanic)	deu	24
Greek (Modern)	Indo-European (Hellenic)	ell	16
Halkomelem	Salishan	hur	19–20
Hungarian	Uralic	hun	16
Imbabura Highland Quichua	Quechuan	qvi	17–18
Indonesian	Austronesian (Malay)	ind	18, 26–27
Japanese	Japonic	jpn	18, 35–36, 62–66
Javanese	Austronesian	jav	82–83
Kannada	Dravidian	kan	17–18
Kinyarwanda	Niger-Congo (Bantu)	kin	19
Lakota	Siouan	lkt	56–62
Mam	Mayan	mam	14
Rawang	Sino-Tibetan	raw	79-82
Russian	Indo-European (Slavic)	rus	15
Sahaptin	Plateau Penutian	yak	4
Tsez	Northeast Caucasian	ddo	71–75
Turkish	Turkic	tur	14, 17, 23, 25
Vengo	Niger-Congo (Grassfields)	bav	17
Wambaya	Australian (Mirndi)	wmb	4
West Greenlandic	Eskimo-Aleut (Inuit)	kal	75–77
Zulu	Niger-Congo (Bantu)	zul	66–70