IMPLEMENTING THE MORPHOLOGY-SYNTAX INTERFACE: CHALLENGES FROM MURRINH-PATHA VERBS

Melanie Seiss
Universität Konstanz

Proceedings of the LFG11 Conference

Miriam Butt and Tracy Holloway King (Editors)

2011

CSLI Publications

http://csli-publications.stanford.edu/

Abstract

Polysynthetic languages pose special challenges for the morphology-syntax interface because information otherwise associated with words, phrases and clauses is encoded in a single morphological word. In this paper, I am concerned with the implementation of the verbal structure of the polysynthetic language Murrinh-Patha and the questions this raises for the morphology-syntax interface.

1 Introduction

The interface between morphology and syntax has been a matter of great debate, both for theoretical linguistics and for grammar implementation (see, e.g. the discussions in Sadler and Spencer 2004). Polysynthetic languages pose special challenges for this interface because information otherwise associated with words, phrases and clauses is encoded in a single morphological word. In this paper, I am concerned with the implementation of the verbal structure of the polysynthetic language Murrinh-Patha and the questions this raises for the morphology-syntax interface.

The Murrinh-Patha grammar is implemented with the grammar development platform XLE (Crouch et al. 2011) and uses an XFST finite state morphology (Beesley and Karttunen 2003). As Frank and Zaenen (2004) point out, a morphology module like this in combination with sublexical rules makes a lexicon with fully inflected forms unnecessary, which is especially important for a polysynthetic language as listing all possible morphological words would be unfeasible, if not impossible. However, this raises the question of the division of work between syntactic grammar rules in XLE and morphological formations in XFST. By looking at different cases of long distance dependencies within the Murrinh-Patha verbal template, this paper contributes to an understanding of the division of work between these two components of grammar.

The paper is structured as follows. Section 2 gives a short overview over the Murrinh-Patha verb and provides some examples of the complexities of the verbal template. Section 3 then outlines the assumed architecture for the morphology-syntax interface. The first part of the section summarizes the theoretical framework put forth by Butt and Kaplan (2002). This framework is then illustrated by the basic outline of the Murrinh-Patha XLE and XFST implementation.

Sections 4 and 5 discuss the details of the implementation of the dependencies in the verbal template. Section 4 deals with the implementation of dependencies concerning subject and object number marking as well as tense marking. I

[†]I would like to thank Rachel Nordlinger who kindly provided me with field data and information about the language. Further thanks go to my supervisor Miriam Butt and the audience of the LFG Conference 2011 for discussions about this topic, and to Jane Simpson and the members of the Language Technology Group at Macquarie University for discussions on previous versions of the talk.

argue that these dependencies should be considered morphological dependencies and should thus be modeled within the morphological component.

In contrast to the purely morphological features discussed in section 4, section 5 introduces dependencies between two lexical-semantic components of the Murrinh-Patha verb. Although the combination of these components is also part of the word formation process, I argue that the dependencies are nevertheless more efficiently modeled in the syntactic component, as their combinatory possibilities depend on syntactic features. Dependencies within a word thus do not have to be modeled in the morphology exclusively. Especially in polysynthetic languages some dependencies may also involve syntactic features and can more efficiently be implemented in the syntax.

2 Dependencies in the Murrinh-Patha Verbal Template

Murrinh-Patha is a non-Pama-Nyungan language spoken around Wadeye in the Northern Territory of Australia. It is a headmarking, polysynthetic language with a very complex verbal template and minimal case morphology on the noun. Due to space limitations only some examples of the dependencies within the verbal template can be discussed here. For a more detailed overview of the complexities of the verbal template see Nordlinger (2010b), and see Blythe (2009) for a more general introduction to the language.

The Murrinh-Patha verbal template can be considered to have nine different slots for verbal stems, agreement markers and incorporated body parts, adverbials or particles. This paper mainly discusses the slots 1, 2, 5, 6 and 8 and their interdependencies.

1	2	3	4	5	6	7	8	9
Class.	SubjN/ Obj	RR	IBP APPL	Lex	TNS	Adv/Prt	SubjN/ ObjN	Adv/Prt

Class:	classifier stem, marked for tense, aspect & subject number
SubjN:	subject number markers for dual & paucal subject
Obj:	object agreement marker
ObjN:	object number marker for dual & paucal
RR:	reflexive / reciprocal marker
IBP:	incorporated body part
APPL:	applicative marker *-ma*
Lex:	lexical stem
TNS:	tense marker
Adv/Prt:	Adverbial / Particle

Figure 1: Murrinh-Patha verbal template (adapted from Blythe 2009)

The semantics of the Murrinh-Patha verb is determined by two morphemes together, the so-called classier stem and the lexical stem. There are 38 different classifier stems. Classifier stems inflect for tense, aspect and subject number and they encode this information in portmanteau forms. They mostly have a generic meaning. In contrast, lexical stems may express more specific meaning and are also noninflecting.

(1) shows some first simple examples.[1] In (1a), the classifier stem SNATCH(9) combines with the lexical stem *rta*, 'hug', while in (1b) the classifier stem 13 combines with the lexical stem *ngkardu* 'see'. (1c,d) show that not every lexical stem can combine with every classifier stem and vice versa. The details of these combinatory possibilities and their consequences for the implementation will be discussed in section 5.

(1) a. *manganta*
mangan-rta
3sgS.SNATCH(9).nFut-hug
'He/she hugged him/her.' (Nordlinger, 2010a)

b. *bamkardu*
bam-ngkardu
3sgS.13.nFut-see
'He/she saw him/her.' (Fieldnotes R. Nordlinger)

c. *mangan - ngkardu

d. *bam - rta

The classifier stems are inflected for tense. However, in all but the non-future tense, a corresponding tense marker has to attach to the classifier stem in slot 6. For example, the tense marker *-nu* has to attach to the future classifier stem form, as in (2a). In (2b), the tense marker *-dha* attaches to the past imperfective form of the classifier stem. (2c) shows that only corresponding tense markers can attach to the classifier stems.

(2) a. ba - ngkardu - nu
1sgS.SEE(13).Fut - see - Fut

b. be - ngkardu - dha
1sgS.SEE(13).PImpf - see - PImpf

c. *ba - ngkardu - dha
1sgS.SEE(13).Fut - see - PImpf

While the tense dependencies can be considered as simple agreement, the relationship between the number information encoded in the classifier stem and the

[1]Because of their generic meaning, classifier stems are glossed either with capitals or a number; lexical stems are glossed according to their meaning in combination with the classifier stem. Abbreviations used in the glosses are: RDP: reduplicated; Fut: Future tense; PImpf: Past Imperfective; du.f: dual female subject; pauc.f: paucal female subject; 1sgDO: 1st singular direct object marker; etc.

separate subject number markers in slots 2 and 8 is more complex. The classifier stems themselves show a three-way number contrast: singular, dual and plural. The singular classifier stem can combine with a separate dual number marker which overwrites the information of the classifier stem, as can be seen in (3a). The dual classifier stem without a separate subject number marker denotes the dual, non-sibling category as in (3b). However, this information can also be overwritten by attaching a paucal number marker as in (3c).

(3) a. bam - ngintha - ngkardu
3sgS.SEE(13).nFut - du.f - see
'They two (non-siblings) saw him/her.'

b. pubamka - ngkardu
3duS.SEE(13).nFut - see
'They two (siblings) saw him/her.'

c. pubamka - ngkardu - ngeme
3duS.SEE(13).nFut - see - pauc.f
'They few (non-siblings) saw him/her.'

As can already be seen in these simple examples, the number markers for dual and paucal subject behave differently in the verbal template. The paucal number marker attaches after the lexical stem (in slot 8), while the dual number marker attaches before the lexical stem (in slot 2).

However, the placement of the dual number marker is more complex as can be seen in the examples in (4). The subject number marker competes with the object markers for slot 2. Thus, if an overt object marker is present, the subject marker has to move to slot number 8 ((4a)). However, as can be seen in (4b), if no overt object marker is present, the subject number marker is disallowed in slot 8.

(4) a. bam - ngi - ngkardu - ngintha
3sgS.SEE(13).nFut - 1sgDO - see - du.f
'They two (non-siblings) saw me.'

b. *bam - ngkardu - ngintha
3sgS.SEE(13).nFut - see - du.f

These quite simple examples already show that the Murrinh-Patha verbal template is quite complex. It involves long distance dependencies as well as constraints which depend solely on the linear ordering within the template, and are not constrained by the functions the morphemes fulfill. Moreover, even more complex cases exist, e.g. with discontinuous object markers, in which case the subject number markers cannot be expressed overtly and the form is ambiguous. These examples, however, are too complex to be treated in detail here and the simpler examples in (1)-(4) suffice to illustrate the analysis. Before going into the details of the analysis, though, the next section lays out the general architecture of the morphology-syntax interface assumed in this paper.

3 Morphology - Syntax Architecture

As morphemes in polysynthetic languages often fulfill similar functions as words fulfill in non-polysythetic language, the dependencies described in the previous section could in principle either be modeled in the syntax or in the morphology. This section describes the general architecture of the morphology-syntax interface assumed in this paper. It first describes the theoretical background of the relationship between morphology and syntax and then shows how such an approach can be realized in a computational implementation.

3.1 General Architecture

The architecture assumed in this paper is a realizational model of morphology that passes on morphological information to the syntax. This morphological information may then trigger syntactic operations.

For a formalization of the architecture, the basic layout of the morhology-syntax interface as proposed by Butt and Kaplan (2002) is used. Butt and Kaplan (2002) define a complex relation *R* to model the interface between morphology and syntax. More precisely, *R* is a relation that "realizes the morphological features of a given f-structure as a string: *f R w*" (Butt and Kaplan, 2002, 3). The complex relation *R* can be decomposed into two relations which Butt and Kaplan (2002) call *Sat* and *D*:

(5) $R = Sat \circ D.$

The relation *Sat* is the satisfaction relation holding between an f-structure and an f-description as already proposed by Kaplan and Bresnan (1982). The description relation *D* models the relation between the f-description and a string. Butt and Kaplan (2002) display *D* as a set of ordered pairs <f-description, sequence> and use (6) as an example for the string /walks/ which is associated with f-descriptions concerning the subcategorization frame of the verb as well as number, person and tense information.

(6) $<$ { (f_1 PRED) = 'walk<(f_1 SUBJ)>', (f_1 SUBJ PERS) = 3,
(f_1 SUBJ NUM) = sg, (f_1 TNS-ASP TENSE) = pres }, /walks/ $>$

As Butt and Kaplan (2002) point out, the relation called *D* is traditionally considered the morphology-syntax interface and how this interface should be formally modeled has been the point of much debate. They propose to model the interface by decomposing *D* further into a lexical relation *L* and a sequence relation *Seq*, which renders (7) as the overall decomposition of *R*.

(7) $R = Sat \circ L \circ Seq$

The lexical relation *L* maps between f-descriptions and what Butt and Kaplan (2002) call description-names (D-names). Examples are given in (8). These D-names are atomic symbols (with arbitrary names, but for convenience mnemonic terms are chosen) and are, in the sequence relation *Seq*, linked to the string, e.g. as in (9).

(8)

walk:	(↑ PRED) = ‘walk<(↑ SUBJ)>’
3:	(↑ SUBJ PERS) = 3
Sg:	(↑ SUBJ NUM) = sg
Pres:	(↑ TNS-ASP TENSE) = pres

(9) < {3, Sg, walk, Pres}, /walks/ >

The described architecture thus associates a string (or phonological word) with an f-structure via a set of D-names and f-descriptions. The use of D-names makes this approach a realizational model, in which the relationship between affixes and their functions can be quite complex.

To summarize, the seqence relation *Seq* maps a set of D-names to a string, determining which strings are possible in a given language. It can thus be considered the morphological part of the relation. The satisfaction relation *Sat* maps between an f-structure and an f-description and is thus part of syntax, while the lexical relation *L* is the mapping between the morphological D-names and the syntactic f-descriptions and can thus be considered the morphology-syntax interface.

By separating these mappings in the described way, this approach is in line with LFG’s general modular architecture. Different mechanisms can be at work on the different levels, and the choice of one model for one level does not necessarily preempt the choice of model for a different level. The following subsection shows how this theoretical architecture can be implemented in a computational XLE / XFST implementation.

3.2 Test Case: XLE / XFST Implementation

In the computational implementation used here as a test case, an XFST finite state morphology (Beesley and Karttunen, 2003) is used in combination with an XLE grammar (Butt et al., 1999; Crouch et al., 2011). XFST morphologies are used in a variety of XLE (ParGram) implementations, and make a lexicon with fully inflected forms unnecessary. This is crucial for a polysynthetic language, as listing all inflected forms for a language like Murrinh-Patha would be unfeasible, if not impossible.

The output of a finite state morphology is a two-sided morphology in which a string is associated with a number of tags to encode information. In (10), the surface form *bamkardu* is associated with the information that this form is made up of the stem *bam*, which is classifier stem number 13 in its 3rd person singular non-future form, an unexpressed 3rd person direct object, and another stem *ngkardu* which is a lexical stem, marked by *+LS*.[2] The relation between the string and the tags (D-names) is thus the instantiation of the sequence relation *Seq*.

(10) bamkardu : bam +class13 +3P +sg +nFut +3sgDO +ngkardu +LS

The morphology output serves as input for XLE and thus needs to be interpretable by the syntax. For this purpose, the D-names need to be associated with f-descriptions. An example of an excerpt of a 'morphological lexicon' instantiating the lexical relation *L* is given in (11). The D-name +class13, for example, passes up the information that the classifier is number 13, the tag +3P that the subject is 3rd person, etc.[3]

(11)	+class13	CLASS	($\uparrow$ CLASSIFIERSTEM) = 13
	+3P	PERS	($\uparrow$ SUBJ PERS) = 3
	+sg	NUM	($\uparrow$ SUBJ NUM) = sg

In order for XLE to interpret these lexical entries, sublexical rules are needed which determine how the constraints for a combination are composed (Kaplan and Newman, 1997). (12) shows a very simple, flat sublexical rule in which the constraints are just passed up.

```
(12)   V   —>   CS: ↑ = ↓
                CLASS: ↑ = ↓
                PERS: ↑ = ↓
                NUM: ↑ = ↓
                ....
```

The sublexical rules in XLE thus have to take up the ordering of the morphology output again.[4] In this way, the sublexical rules mirror the morphology output. While this may seem to be an unnecessary complication of the implementation, it offers the possibility of testing various ways of implementing morphologically complex words. The dependencies in the Murrinh-Patha verb, for example, could be modeled either in the XFST morphology or in XLE in the sublexical rules. In the remainder of this paper I show that it makes sense to treat morphological dependencies in the morphology while other dependencies can be left for the syntax.

[2]The details of the implementation will be explained in subsection 4.1.

[3]This is a very simple example in which a morphological feature only triggers one syntactic interpretation. However, the Murrinh-Patha system is more complex than that. For details of the analysis, for example of the number features, see Nordlinger (2010a).

[4]As T.H. King pointed out, this does not have to be the case necessarily. Instead, variables of the form "{stem | affix}+" could be used. Most ParGram grammars, however, use detailed sublexical rules, as e.g. exemplified in (12), to allow for using the same tag in different sublexical rules.

4 Morphological Interdependencies

In this section the implementation of tense marker dependencies and the interdependencies of subject and object markers is discussed. I show that it is not viable to model these dependencies within XLE, while it is possible in XFST.

This may seem surprising at first glance as actually, the formalisms of XLE and XFST are equivalent, as finite state automata can be translated into context-free grammars and vice versa. However, I am concerned with the interplay of these two formalisms. The combination of both formalisms makes it apparent that it is theoretically desirable to make a distinction between morphological and syntactic dependencies within a word and consequently to model these dependencies in different modules.

The first subsection discusses the implementation of the subject and object markers in XLE and argues that it is not theoretically desirable to model the complex interdependencies in XLE. The second subsection then shows how these dependencies can be modeled in XFST which allows for a very simple XLE sublexical rule.

4.1 Tense, Subject and Object marking in XLE

The architecture of the morphology-syntax interface described in the previous section offers the possibility to model the dependencies found in the Murrinh-Patha verb in XLE with the help of sublexical rules. This may seem feasible as word formation processes for a polysynthetic language may be similar to the formation of phrases in non-polysynthtic languages. However, in this subsection I discuss that modeling complex morphological dependencies between subject and object markers in XLE faces various problems.

As was discussed in section 2, the dependencies involving subject and object marking involve 3 different verbal template slots, i.e. the classifier stem form (slot 1) as well as slot 2 and slot 8, which host the special markers for subject number and direct and indirect object person and number marking.

For reasons of space, only the case for the singular classifier stem form will be discussed. The relevant data is repeated in (13) and shows the interplay of the placement of the subject number marker and the direct object marker. The dual subject number marker competes with overtly expressed direct object markers for slot 2. Thus, when no direct object marker is expressed overtly as in (13a), the dual subject number marker is in slot 2. When a direct object marker is expressed overtly, the subject number marker has to move to slot 8 ((13b)). However, the dual subject number marker is ungrammatical in slot 8 when slot 2 is not filled ((13c))

(13) a. bam-nintha-ngkardu :
bam +class13 +3P +sg +nFut **+3sgDO** **+du.m.Nsibl.S** +ngkardu +LS

b. bam-ngi-ngkardu-ngintha:
bam +class13 +3P +sg +nFut **+1sgDO** +ngkardu +LS **+du.f.Nsibl.S**

c. *bam-ngkardu-ngintha:
bam +class13 +3P +sg +nFut **+3sgDO** +ngkardu +LS **+du.f.Nsibl.S**

Because the dependencies are subject to the linear ordering within the verbal template, i.e. they do not solely depend on whether the markers are present or not, the only possibility for modelling these dependencies in XLE is in the sublexical rules. However, the dependencies are also long-distance, which means one needs to keep track of what choices have been made in the other template slots.[5]

This can be achieved by introducing so-called "CHECK" features in the f-structure. The sublexical rule in (14), for example, models the dependency between subject and object markers for singular classifier stems. If the object is overtly expressed, i.e. non-3rd person singular, the subject number marker can only be expressed in slot 8, after the lexical stem. The subject marker can only be expressed before the lexical stem, i.e. in slot 2, if the object marker is not overtly expressed, i.e. if the object is 3rd person singular or if it is an intransitive verb.

(14) V —> CS: (↑TAM TENSE) =c non-fut;
CLASS PERS NUM TENSE
{ (DO: (↑OBJ NUM) =c sg (↑OBJ PERS) =c 3)
(↑ CHECK _DO) = 3sg
|
DO: {(↑ OBJ NUM) =c {dual | paucal | pl } |
(↑ OBJ NUM) =c sg (↑ OBJ PERS)=c 1 |
(↑ OBJ NUM) =c sg (↑ OBJ PERS) =c 2 }
(↑ CHECK _DO) = non3sg }
(SNUM2: **(↑ CHECK _DO) =c 3sg**)
....
LexStem
(SNUM2: **(↑ CHECK _DO) =c non3sg**)

This implementation has various disadvantages. First, there is no principled way of talking about the overtly expressed object markers in the syntax. The notion of whether a marker is present or not is morphological: on this level only the functions of the markers are left. We thus have to tie the two alternatives to having a 3rd singular direct object or all other cases, which have to be listed individually.

Second, using CHECK features undesirably leads to an f-structure which is crowded with information that is not important functionally, but only serves to keep track of the form of the lexical entry. These features are standardly used within the ParGram group for XLE grammar writing, for example for the implementation of auxiliary verb constructions, as has been discussed by Butt et al. (2004). Butt et al. (2004) propose a separate m-structure to keep track of these morphological forms to avoid these features in the f-structure. This m-structure, however, cannot be considered a complete morphological structure, and proposals to expand it to a real

[5]Alternative implementations also exist, i.e. one could list all possible combinations of sublexical rules. However, this would be unfeasible as it would involve many different sublexical rules for all different combinations (tense, subject number etc.).

morphological structure, as e.g. proposed by Frank and Zaenen (2004), lead to an unnecessary partial reduplication of the f-structure in the m-structure.

For periphrastic expressions such as auxiliary verb combinations, these CHECK features are sometimes inevitable because they model the dependencies between different words. However, in the Murrinh-Patha case the CHECK features used in (14) model dependencies within a word. For such cases modeling the dependencies directly in the morphology results in a much cleaner division of work between the morphology and syntax, and a cleaner division between morphological and syntactic features. Thus, the next section describes how these dependencies can be modeled within the morphology using XFST.

4.2 Tense, Subject and Object marking in XFST

The last section showed that modeling the dependencies of the Murrinh-Patha verbal template within the sublexical rules in XLE is inelegant. This section now discusses how the dependencies can be modeled in the morphology, i.e. with XFST. I first provide a short introduction to XFST and the basic mechanisms, and then explain how the long distance dependencies can be modeled with the help of flag diacritics.

The concept of finite state morphology was developed in the 1980s as a tool for the computational morphological analysis of natural language. It combines ideas of sequenced phonological rewrite rules with two-level morphology (Koskenniemi, 1983). For a detailed historic overview and a formal description of the formalisms at work see Beesley and Karttunen (2005).

Different tools exist which allow the implementation of finite state morphologies. In the implementation of Murrinh-Patha discussed here, XFST in connection with LEXC (Beesley and Karttunen, 2003) is used. The discussion of the implementation of Murrinh-Patha verbal templates will only be concerned with the creation of the verbal lexicon with LEXC. Other questions such as the modeling of the phonological changes that apply are left undiscussed.

LEXC uses two-sided continuation classes to model the concatenation of strings. The mechanism is best explained with an example. (16) describes a network that produces the output in (15). The colon separates the level of the surface form *bamngkardu* on the lower side and the morphological information this surface form is associated with.

(15) bam +class13 +3P +sg +nFut +3sgDO +ngkardu +LS : bamngkardu

(16)
```
Lexicon ROOT
bam+class13+3P+sg+nFut:bam    OBJECT;

Lexicon OBJECT
+3sgDO:0                      LEX;

Lexicon LEX
+ngkardu+LS:ngkardu           #;
```

LEXC uses continuation classes which are implemented as so-called lexicons. The first lexicon is called ROOT, it comprises all possible first morphemes of a word. In this lexicon, the classifier string *bam* is associated with the morphological information that it carries (*bam+class13+3P+sg+nFut*). The right side of the lexicon entry specifies which lexicon is used next. In (16), *bam* can be concatenated with objects from the lexicon OBJECT, which in this case only contains the 3rd person direct object marking which is not overtly realized (noted as 0). This combination combines with items from the lexicon LEX, which contains the lexical stem *ngkardu*. The hash key marks the end of a word.

In the actual implementation of Murrinh-Patha verbs, the lexicon ROOT contains all forms of the 38 classifier stems, and a large number of different lexical stems are contained in the lexicon LEX. The other template slots are implemented with the help of lexicons in a similar way.

Dependencies between neighboring lexicons can be easily modeled by specifying different continuation classes, i.e. entries of one lexicon do not have to lead to the same next lexicon. However, most dependencies in the Murrinh-Patha verbal template are long-distance, which is very difficult to model just with the concatenation described above.

For long distance dependencies, flag diacritics are used in the implementation of the Murrinh-Patha verbal template. Flag diacritics are special entities in XFST which add a kind of "short term memory" to keep track of what choices have been made before. Thus, as Beesley and Karttunen (2003, 341) explain, normally, "the transition from one state to the next depends only on the current state and the next input symbol". Using flag diacritics, however, allows one to keep track of those choices, so that certain transitions can also be constrained by choices made earlier.

In the implementation, flag diacritics can be recognized by two surrounding @-symbols. After the first @-symbol, an operator is followed by a feature-value pair, each separated by periods. Different operators exist, i.e. U(nification), P(ositive) (Re)setting, R(equire) test, D(isallow) test etc. The names of the features and values can be chosen arbitrarily, but for convenience, morphological features and values have been chosen.

As a first simple illustration of the use of flag diacritics, the long distance dependency between the tense marking on the classifier stem and separate tense markers in slot 6 will be used. For all tenses but the non-future tense, tense markers in slot 6 are obligatory. The relevant examples are repeated in (17). In example (17), *bam* is the non-future form of the classifier stem 13 while *ba* is the future form of the corresponding classifier. The future form has to combine with the future tense marker *-nu* (tagged as +Fut2) as can be seen in (17b); it is ungrammatical without *-nu* ((17d)). On the other hand, *-nu* cannot attach to the non-future classifier stem form ((17c)).

(17) a. bam-ngkardu : bam +class13 +3P +sg +nFut +3sgDO +ngkardu +LS
b. ba-ngkardu-nu : ba +class13 +3P +sg +Fut +ngkardu +LS +Fut2
c. *bam-ngkardu-nu :

bam +class13 +3P +sg +nFut +3sgDO +ngkardu +LS +Fut2

d. *ba-ngkardu : ba +class13 +3P +sg +Fut +ngkardu +LS

This interplay can be modeled with the help of P- and R-type flag diacritics as in (18). In the lexicon ROOT, the classifier stem forms are associated with the classifier number information as well as person, subject number and tense information. The flag diacritics "@P.Tense.nFut@" and "@P.Tense.Fut@" remember the choices made for the tense values. When the corresponding tense markers are attached in slot 6, *-nu* can only attach to a future classifier stem form, i.e. this choice requires that the feature "Tense" has been set to the future value before. Similarly, the first line in the lexicon TENSE specifies that no tag is only possible if the value of the feature "Tense" has been set to "nFut" before.

(18)
```
Lexicon ROOT
bam ... +nFut@P.Tense.nFut@:bam@P.Tense.nFut@    LEX;
ba ... +Fut@P.Tense.Fut@:ba@P.Tense.Fut@         LEX;
....
Lexicon TENSE
@R.Tense.nFut@                                   #;
+2Fut@R.Tense.Fut@:nu@R.Tense.Fut@               #;
```

These dependencies for tense markers are quite simple examples of long distance dependencies. However, flag diacritics also allow the modeling of complex long distance dependencies such as the subject number and object marker dependencies which are dependencies between three different verbal template slots. (19) provides an example of such a complex interplay by modeling the facts displayed by the examples in (13). It is thus the XFST alternative to the XLE implementations in (14).

(19)
```
Lexicon ROOT
bam+class13...+sg@P.NUM.sg@..:bam@P.NUM.sg@                 SLOT2;

Lexicon SLOT2
@P.SMark.no@                                                   RR;
+1sgDO:ngi                                                     RR;
+du.m.Nsibl.S@P.SMark.pres@@R.Num.sg@
        :nintha@P.SMark.pres@@R.Num.sg@                        RR;
...
Lexicon SLOT8
+du.m.Nsibl.S@D.SMark.pres@@D.SMark.no@@R.Num.sg@
          :nintha@D.SMark.pres@@D.SMark.no@@R.Num.sg@           #;
```

The excerpt in (19) models the dependencies between subject number and object markers for the singular classifier form. In the lexicon ROOT, the classifier form *bam* is associated with the singular form of classifier 13, and this choice is marked by the P-type flag diacritic, i.e. it remembers that the value for the number feature singular has been set positively.

In the lexicon SLOT2, three different choices are possible. In the first case, nothing is attached. This is for example the case for intransitive verbs with singular subjects. However, the system has to remember that nothing has been attached in this slot, which is implemented with the flag diacritic @P.SMark.no@. Alternatively, an overtly expressed object marker can attach in slot 2, i.e. the marker for the 1st person singular direct object marker *-ngi*. As a third choice, the dual masculine non-sibling subject number marker *-nintha* can attach in slot number 2. However, *-nintha* can only attach if the classifier stem form is singular, which is modeled by the flag diacritic @R.Num.sg@, which requires the value of the number feature to have been positive before. In this case, the flag diacritic @P.SMark.pres@ tells the system to remember that the dual subject marker is present in slot 2.

The lexicon SLOT8 then takes care of all the possible choices. Thus, the dual subject number marker can only attach in slot 8 if it is not present in slot 2. This dependency is modeled by the flag diacritic @D.SMark.pres@ which disallows this choice if the value of the feature SMark has been set to “pres(ent)” before. Secondly, the dual number marker can only attach in slot 8 if slot 2 is not empty, i.e. this choice is disallowed if the value of the SMark has been set to “no” before. And thirdly, as has been already discussed before, the classifier stem has to be in its singular form.

The combination of different flag diacritics thus models the dependencies between singular classifier forms, dual subject number markers and object markers in slots 2 and 8. It is similar to modeling the dependencies in the alternative XLE implementation in (14) involving CHECK features. However, the implementation in XFST has a range of advantages over the corresponding XLE implementation.

First, flag diacritics model the dependencies within the morphology and are therefore invisible to the syntax. In contrast to the CHECK features, they do not show up in the f-structure or need to be put in a separate m-structure.

Second, the features modeled here are morphological features and it was very difficult to address these in XLE. Flag diacritics, however, make it easy to talk about separate morphological features on the one hand, for example as in the @P.Num.sg@ flag which picks out the number feature from the classifier stem form. On the other hand, it is easy to combine features and remember the choice of a combination of features by flag diacritics, i.e. as in the case of the flag diacritic @P.SMark.pres@, which remembers that the subject number marker is dual, masculine and non-sibling. In a way, however, it just remembers that this marker has been present in slot 2, i.e. it is bound to the appearance of the morpheme, and not to the features it represents. This is not possible in XLE as it assumes the strict lexicalist hypothesis that the internal structure of words is not visible to the syntax.

To sum up, in this section different possibilities for the implementation of long distance dependencies in the Murrinh-Patha verbal template have been discussed. I argued for a treatment of the tense marker and the subject number and object marker dependencies in XFST as the dependencies are morphological and should therefore be modeled in the morphology. The features in question are morphological features, and, more importantly, the dependencies are influenced, to a large degree, not only by the features realized by the markers, but by the linear ordering of the markers in the verbal template.

The next section is concerned with a different set of dependencies, i.e. the dependencies between the classifier stem and the lexical stem. In contrast to the dependencies discussed above, these dependencies seem to be syntactic (or even semantic) in nature and should thus be treated differently from the morphological dependencies.

5 Classifier plus lexical stem combinations

While the previous section dealt with purely morphological dependencies in the Murrinh-Patha verb such as dependencies between tense or number features, this section discusses the combination of lexical stem and classifier stem. As has been argued by Seiss and Nordlinger (2010), the combinations can be considered complex predicates in which both stems contribute part of the meaning and the combination determines the syntactic information. Although a lexical stem and classifier stem form one word together and their combination is thus a morphological formation, I argue that it makes more sense to model the dependencies between these stems in the syntax, as their combinatory possibilities are determined by syntactic features.

This section is divided into two parts. First, the different combinatorial possibilities of lexical and classifier stems will be discussed briefly. I show that while some regularity exists when valency matching is assumed, many combinations are in fact lexicalized or semi-productive, so that we need to associate the different combinations with distinct syntactic lexicon entries. This leads to the analysis of these dependencies within XLE, which will be discussed in the second part of the section.

5.1 Empirical basis

In the paper so far, only one simple example of a classifier plus lexical stem combination has been used to illustrate the morphological complexity of the verbal template. However, many different combinations of classifier and lexical stems exist. This section presents the main patterns found in the combinations and discusses their syntactic properties.

Different lexical stems can combine with different classifier stems, and vice versa. (20) shows an example of the same lexical stem (or, more precisely, its reduplicated form) in combination with two different classifier stems, STAND(3) and HANDS(8). Although the combinations are formed with the same lexical stem, they differ in their valency. (20a) is intransitive while (20b) is transitive.

(20) a. ngirra - dharday - nu
3sgS.STAND(3).Fut - down - Fut
'I'll descend straight down.' (Street and Street, 1989)

b. nanthi karlay mam-dhardarday
NC fishing net 3sgS.HANDS(8).nFut-down(RDP)
wurran
3sgS.GO(6).nFut
'He continually lets the fishing net down.' (Street and Street, 1989)

This alternation can be explained by assuming that the valency of the complex predicate usually follows the valency of the classifier stem. Classifier stems can be divided into intransitive and transitive classifier stems, as well as Reflexive/ Reciprocal (RR) classifier stems. Determining the valency of lexical stems is more difficult; however, the valency of many lexical stems can be derived based on their lexical semantic meaning and the valency patterns in their combinations with classifier stems.

In many cases, it seems that the valency of the classifier and the lexical stem match. However, other patterns are also possible. (21) shows three different possibilities with intransitive classifier stems. In (21a), the classifier stem and lexical stem are both intransitive, and the resulting combination is intransitive as well. This seems to be a case of valency matching.

In contrast, in (21b), the transitive lexical stem *lerrkperrk* combines with the intransitive classifier stem SIT(1). The combination is intransitive with a resultative meaning in which only the patient is expressed. Thus, in this case the valency of the combination also follows the valency of the classifier stem.

This, however, is not always the case. (21c) is formed with the intransitive classifier stem BE(4) combined with *gurdugurduk* 'drink'. The resulting combination is transitive, and it seems that BE(4) only contributes some aspectual meaning.

(21) a. dim - karrk
3sgS.SIT(1).nFut - cry
'He's crying.' (Street and Street, 1989)

b. dim - lerrkperrk
3sgS.SIT(1).nFut - crush
'It's smashed.' (Seiss and Nordlinger, 2010)

c. kura patha kanam - gurdugurduk
NC:water good 3sgS.BE(4).nFut - drink(RDP)
'He continually drinks water.' (Street and Street, 1989)

The examples in (20) and (21) show that the subcategorization frames of the verbs cannot be predicted generally. While there is a certain regularity for valency matching, other factors also play a role. Certain causative lexical stems may combine with SIT(1) to form resultative verbs, as in (21b). Some intransitive classifier stems may combine with certain lexical stems and then only contribute aspectual information. Thus, the subcategorization frames of the combinations have to be listed in the lexicon, more precisely, in the XLE lexicon which determines the syntactic information.

Apart from the subcategorization frame, classifier and lexical stem combinations also determine other syntactic or semantic information together, for example in their interpretation of RR classifier stems. (22a) shows the Reflexive/Reciprocal classifier 15 which is the corresponding RR classifier for classifier 13. In combination with the lexical stem *ngkardu*, it forms a reflexive verb. This is the productive use of the RR classifier stems.

However, other examples such as (22b) exist in which the combination is ambiguous between a resultative reading and the productive reflexive reading. Finally (22c) is a purely lexicalized version in which only the resultative reading is available.

(22) a. bem - ngkardu
1sgS.15.nFut - see
'I saw myself.' (Nordlinger 2008)

b. mem-let
3sgS.HANDS:RR(10).nFut - stick
'It's already stuck up (e.g. on the wall).' (Nordlinger, 2011)
'It stuck itself up on the wall.' (Rachel Nordlinger, pc)

c. nhem - nham
1sgS.POKE:RR(21).nFut-fear
'I'm afraid.' (Seiss and Nordlinger, 2010)

Finally, classifier and lexical stems together determine how the object marker is interpreted. Murrinh-Patha has a considerable amount of so-called impersonal verbs (Walsh, 1987) in which the direct object marker actually denotes the subject, as can be seen in (23).

(23) a. pan - ngi - ngkawerr
3sgS.23.nFut - 1sgDO - terrify
'I'm terrified.' (Walsh, 1987)

b. dem - ngi - ralal?
3sgS.POKE:RR(21)nFut - 2sgDO - thirsty
'Are you thristy? ' (Nordlinger, 2011)

Summing up briefly, classifier and lexical stems together determine a range of syntactic features, such as subcategorization frames, and reflexivity and reciprocality, aspectual information such as the resultative reading and the mapping from thematic roles to grammatical functions as for the impersonal verbs. This makes detailed lexical entries tied to syntactic features necessary.

5.2 Modeling classifier plus lexical stem combinations

In the previous section I argued for the need of lexical entries for classifier and lexical stem combinations with detailed syntactic information. This section discusses the different possibilities of modeling the dependencies and the consequences for the division of work between syntax and morphology. I show that while it would be possible to model the dependencies in the morphology, it is more efficient and theoretically elegant to model them in the syntax.

(24) shows an excerpt from the XFST lexicons for the classifier and lexical stems used in (20). To model the long distance dependencies, flag diacritics are used to remember which classifier stem has been chosen. For example, the lexical stem *dharday* used in the examples in (20) can combine with the classifier stems 3 or 8 (among others), i.e. in the XFST implementation *dharday* needs to be listed with flag diacritics requiring the classifier stem to be either 3 or 8.

(24) Lexicon Classifer Stems
ngirra @P.CLASS.3@ : ngirra +class3 @P.CLASS.3@
mam @P.CLASS.8@ : mam +class8 @P.CLASS.8@
...
Lexicon Lexical Stems
dharday @R.CLASS.3@ : dharday +LS @R.CLASS.3@
dharday @R.CLASS.8@ : dharday +LS @R.CLASS.8@

While implementing the dependencies in XFST like this is possible, it faces various disadvantages over the implementation of these dependencies in the syntax. As can be seen, implementing the dependencies in XFST requires multiple entries for lexical stems when they can combine with various classifier stems. Moreover, the dependencies are stipulated as no explanation for the restrictions can be found in the morphology.

In contrast, when modeling the dependencies within XLE, the restrictions are tied to the different subcategorization frames and other syntactic or semantic information. For example, the morphology provides the output in (25) for the verbs in the examples in (20).

(25) a. ngirradhardaynu : ngirra+class3+1P+sg+Fut+dharday+LS+Fut2
b. mamdharday : mam+class8+3P+sg+nFut+3sgDO+dharday+LS

This output is interpretable by XLE when the lexical entries for the tags of the classifier stems are given as in (26). In this case, they just pass up the number of the classifier stem. These combine with the lexical entries for the lexical stem, for example as in (27) for the lexical stem *dharday*. This entry then specifies that *dharday* only needs a subject when combined with classifier stem 3 while *dharday* with classifier stem 8 needs a subject and an object.

(26) +class3 CS ($\uparrow$ ClassifierStem) = 3.
+class8 CS ($\uparrow$ ClassifierStem) = 8.

(27) +dharday LS { ($\uparrow$ PRED) = 'down < ($\uparrow$ SUBJ) >'
($\uparrow$ ClassifierStem) =c 3
| ($\uparrow$ PRED) = 'down < ($\uparrow$ SUBJ)($\uparrow$ OBJ) >'
($\uparrow$ ClassifierStem) =c 8. }

Even more syntactic information is needed for the interpretation of the impersonal verbs. As can be seen in (28), the morphological information is the same as for any other classifier plus lexical stem combination. However, the lexicon entry within XLE needs to specify that *ngkawerr* in combination with classifier 23 needs a subject only and that this combination is an impersonal verb.

(28) panngingkawerr : pan+class23+3P+sg+nFut+1sgDO+ngkawerr+LS

```
(29)   ngkawerr   LS   (↑ PRED ) = 'terrify < ( ↑ SUBJ) >'
                       (↑ ClassifierStem) =c 23
                       (↑ Impersonal_Verb) = +.
```

The information (↑ Impersonal_Verb) = + ensures that the morphological tag +1sgDO is interpreted as providing information about the subject, not as an object marker. For this purpose, the lexical entries for the tags providing information about the subject and object need to be complex. The tags for subject information, for example for singular number as in (30), only optionally provide information about the subject, they do not provide any information in the case where the lexical and classifier stem combination is an impersonal verb. The tag for the "object" marker, similarly, may provide information about the subject or the object, as in (31).

```
(30)   +sg   NUM   {   (↑ SUBJ NUM) = sg
                       (↑ Impersonal_Verb) ¬= +
                   |   (↑ Impersonal_Verb) =c +     }

(31)   +1sgDO   OBJ   {   (↑ OBJ PRED) = 'PRO'
                          (↑ OBJ NUM) = sg
                          (↑ OBJ PERS) = 1
                      |   (↑ SUBJ PRED = 'PRO'
                          (↑ SUBJ NUM) = sg
                          (↑ SUBJ PERS) = 1          }
```

This is an example of a complex morphology-syntax interface, or complex lexical relation *L* in Butt and Kaplan's (2002) terms. For a motivation of this analysis and more details see Nordlinger (2010a).

Similar syntactic lexion entries can be defined for the examples involving RR classifier stems or other lexicalized combinations of classifier and lexical stems, as discussed in the previous section.

To sum up, while it would be possible to implement the dependencies between classifier and lexical stems in XFST, these dependencies are stipulative. As the combinations have to be listed in the XLE lexicon anyway, modeling the dependencies in XFST is unnecessary. Thus, although the combination of classifier and lexical stem is part of the word formation process, restricting the possible combinations is better left for the syntax, as the combinations are restricted by syntactic and semantic features.

6 Conclusion

This paper argued for a different treatment of morphologically and syntactically motivated dependencies in the Murrinh-Patha verbal template. Dependencies which encode morphological features and depend on the linear order and the template slots are modeled in the XFST morphology. This avoids the use of morphological form features in the f-structure and is thus true to LFG's lexicalist hypothesis.

On the other hand, the dependencies between classifier and lexical stems are modeled in the syntax. They do not depend on linear order, but rather on syntactic features such as valency, aspect, etc. Moreover, the semantic meaning of a verb is determined jointly by the classifier stem and the lexical stem so that both components need to be present in the syntax in order to be able to be passed on to the semantics. This shows that just because some morphemes combine to form a word, their restrictions are not always best treated in the morphology.

The division of work between morphology and syntax is made possible by the sophisticated morphology-syntax interface assumed in this paper. Due to the modular architecture, dependencies within a word can either be dealt with in the syntax or the morphology. This allows us to treat the phenomena in a computationally efficient and theoretically elegant way.

References

Beesley, Kenneth R. and Karttunen, Lauri. 2003. *Finite State Morphology*. Stanford: CSLI Publications.

Beesley, Kenneth R. and Karttunen, Lauri. 2005. 25 Years of Finite State Morphology. In Antti Arppe, Lauri Carlson, Krister Lindén, Jussi Piitulainen, Mickael Suominen, Martti Vainio, Hanna Westerlund and Anssi Yli-Jyrä (eds.), *Inquiries into Words, Constraints and Contexts: Festschrift for Kimmo Koskenniemi on his 60th Birthday*, pages 71 – 84, Stanford: CSLI Publications.

Blythe, Joe. 2009. *Doing Referring in Murriny Patha conversation*. Ph. D.thesis, University of Sydney.

Butt, Miriam and Kaplan, Ron. 2002. The Morphology-Syntax Interface in LFG, talk presented at the LFG Conference 2002, Athens.

Butt, Miriam, King, Tracy Holloway, Niño, María-Eugenia and Segond, Frédérique. 1999. *A Grammar Writer's Cookbook*. Stanford: CSLI.

Butt, Miriam, Niño, María-Eugenia and Segond, Frédérique. 2004. Multilingual Processing of Auxiliaries within LFG. In Louisa Sadler and Andew Spencer (eds.), *Projecting Morphology*, pages 11 – 23, Stanford: CSLI Publications.

Crouch, Dick, Dalrymple, Mary, Kaplan, Ron, King, Tracy, Maxwell, John and Newman, Paula. 2011. XLE Documentation, URL: http://www2.parc.com/isl/groups/nltt/xle/doc/xle_toc.html.

Frank, Anette and Zaenen, Annie. 2004. Tense in LFG: Syntax and Morphology. In Louisa Sadler and Andew Spencer (eds.), *Projecting Morphology*, pages 23 – 66, Stanford: CSLI Publications.

Kaplan, Ron and Newman, Paula. 1997. Lexical Resource Reconciliation in the Xerox Linguistic Environment. In Dominique Estival, Alberto Lavelli, Klaus Netter and Fabio Pianesi (eds.), *Proceedings of the ACL Workshop on Computational Environments for Grammar Development and Linguistic Engineering, Madrid*, pages 54 – 61.

Kaplan, Ronald M. and Bresnan, Joan. 1982. Lexical-Functional Grammar: A Formal System for Grammatical Representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*, pages 173 – 281, MIT Press.

Koskenniemi, Kimmo. 1983. Two-level Morphology: A General Computational Model for Word-form Recognition and Production, Publication 11, University of Helsinki, Department of General Linguistics, Helsinki.

Nordlinger, Rachel. 2010a. Agreement Mismatches in Murrinh-Patha Serial Verbs. In Yvonne Treis and Rik De Busser (eds.), *Selected Papers from the 2009 Conference of the Australian Linguistic Society*.

Nordlinger, Rachel. 2010b. Verbal Morphology in Murrinh-Patha: Evidence for Templates. *Morphology* 20(2), 321–341.

Nordlinger, Rachel. 2011. Reciprocals in Murrinh-Patha, under review.

Sadler, Louisa and Spencer, Andew (eds.). 2004. *Projecting Morphology*. Stanford: CSLI Publications.

Seiss, Melanie and Nordlinger, Rachel. 2010. Applicativizing Complex Predicates: A Case Study from Murrinh-Patha. In Miriam Butt and Tracy Halloway King (eds.), *Proceedings of the LFG10 Conference, University of Ottawa, Canada*, pages 416 – 436.

Street, Chester and Street, Lyn. 1989. Murrinh-Patha Vocabulary, electronic MS Word file, ms, Darwin N.T.

Walsh, Michael. 1987. The impersonal verb construction in Australian Languages. In Ross Steele and Terry Threadgold (eds.), *Language Topics: Studies in honour of Michael Halliday*, pages 425 – 438, Amsterdam: John Benjamins.