When can a prefix block a root portmanteau?

Demirok, Ömer
Boğaziçi University

Abstract. I provide evidence from Laz that the lexicalisation algorithm in Nanosyntax makes the correct predictions about when a prefix may or may not block a root portmanteau. I argue that prefixes in Laz fall in three classes. First, polarity and agreement prefixes are correctly predicted to not block the root portmanteaus, for they seem to be exponing features higher than the aspect features that the portmanteaus lexicalise. Second, spatial prefixes are also correctly predicted not to block portmanteaus since they originate very low in the structure as complements but end up as specifiers. Since they are specifiers, they can be moved out to give way to root portmanteaus. Finally, pre-root vowels in Laz are projecting complex left branches, exponing features that are in the way of the root portmanteaus, i.e. lower than aspect features. Given their projecting nature, they cannot be moved out, rendering any portmanteau that reaches up to higher features unusable.

1 Introduction

Cross-linguistically, contextual allomorphy is subject to a strict form of locality (Božič 2018). In realizational and morpheme-based models of morphology like Distributed Morphology [DM] (Halle & Marantz 1993) and Nanosyntax [Nano] (Starke 2009; Caha 2009), contextual allomorphy entails that exponents *compete* for insertion into a syntactic node, which is resolved according to some metric designated by the respective models (Caha 2018; Gouskova & Bobaljik 2019).

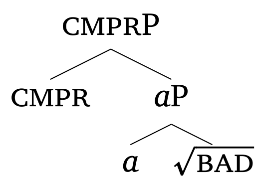
Within the DM approach, there is a debate concerning whether the relevant metric of locality should be stated in terms of linear adjacency (Embick 2010; Ostrove 2018) or structural contiguity (Bobaljik 2012). There are also proposals that argue for relaxing the locality conditions in various ways (Merchant 2015; Smith et al. 2019; Choi & Harley 2019). This debate naturally arises within DM because it is a theoretically open question what locality restrictions determine which DM rules like (1) are licit and which are illicit. In particular, we have the question of how far apart (in syntax or linearly) α and β can be, where α is the node of insertion and β is the *conditioning* context. A concrete example is given in (2).

(1) $\alpha \longleftrightarrow [\ldots] \ / \ \_\ \beta$

(2) a. $\sqrt{\text{BAD}} \longleftrightarrow$ 'worse' / _ CMPR
b. CMPR $\longleftrightarrow \emptyset$ / 'worse' _

The Nano approach, on the other hand, dispenses with contextual allomorphy rules of the kind in (1) and defends the idea that all instances of contextual allomorphy should be analyzed as portmanteau. A fundamental assumption in Nano is *phrasal spell-out*, which allows (in fact requires) a lexical item to be matched with a constituent built in syntax (Starke 2009; 2018). Phrasal spell-out allows a case like in (2) to be explained as a portmanteau which also spells out 'the conditioning context' in the traditional sense, as shown in (3).

(3) 'worse' $\Longleftrightarrow$



More recently, how *linearly uninvolved* morphemes affect allomorph selection has become a question of interest with respect to its potential to adjudicate between structural contiguity and linear adjacency views on allomorph selection (Choi & Harley 2019). The linear adjacency view predicts that linearly uninvolved morphemes will have no effect on the allomorph selection whereas the structural contiguity view predicts that they will have an effect on the allomorph selection as long as they are structural interveners. While structural contiguity may be stated as a stipulation on allomorph selection in DM, a strict form of structural contiguity follows as a theorem in Nano. In this paper, I will couch the entire discussion in Nano. However, the status of linearly uninvolved morphemes has the potential of empirically informing both approaches.

This brief contribution has the modest goal of presenting a case study of root allomorphy patterns in Laz (endangered, South Caucasian). The root allomorphy patterns in this language can answer the above-mentioned empirical question about linearly uninvolved morphemes. I will argue that the root allomorphs in Laz are amenable to a Nano analysis where they are portmanteau exponents that also lexicalise features that are normally realized as suffixes. Furthermore, the language also has prefixes, some of which turn out to block the insertion of these root portmanteaus. I argue that this typology is correctly predicted if prefixes may be born in two ways, as proposed in Starke (2018).

2 Facts on root allomorphy in Laz

We will be concerned with aspect/tense related allomorphy in verbal roots in Laz. Therefore, some brief remarks regarding the verbal complex in Laz are in order.

As shown in (4), the Laz verb has suffixes for voice, aspect, tense, agreement related features; and prefixes for polarity, spatial, and agreement features. In addition, the root is often preceded by a pre-root vowel (PRO) {i-, u-, o-, a-}. PRVs seem to mark fairly low, VP-related features related to voice, applicativization, and root-modality. The verbal complex can get as complex as (5) or can quite simple as in (6).

(4) ... POL + DIR + AGR + PRV + $\sqrt{\ }$ + CAUS + ASP + AUX + TNS + AGR ...

(5)

va-	ce-	v-	o-	çum	-ap	-am	-t’	-i	-t
NEG-	DOWN-	1-	CAUS-	beat	-CAUS	-PROG	-AUX	-PST.1	-PL

‘We weren’t letting them beat him.’

(6) t'ax -es
break -PST.3PL
'They broke it.'

2.1 The basics of tense aspect marking

Laz marks progressive aspect overtly as in (7) but lacks any other overt aspect marking. Tense markers are portmanteaus for tense + agreement. I take past forms as in (8) that lack progressive marking to be perfective.

(7) t'ax **-um** -an
break -PROG -PRS.3PL
'They are breaking it.'

(8) t'ax -es
break -PST.3PL
'They broke it.'

Laz can also build past progressive forms as shown in (9). Notably, an invariant auxiliary -t' appears between the progressive and the tense + agreement markers.[1]

(9) t'ax **-um** -t' -es
break -PROG -AUX -PST.3PL
'They were breaking it.'

The progressive marking presents an important complication, which I will have to gloss over in this paper. The progressive aspect exhibits root-dependent allomorphy, where roots select one of these three markers: {-um, -am, -ur}.[2] Notably, the class distinctions are neutralized when the root directly occurs with the past tense marker, which is invariant across these root classes.

(10) a. zd -es
pull -PST.3PL
'They pulled it.'

b. zd **-am** -an
pull -PROG -PRS.3PL
'They are pulling it.'

1. I gloss -t' as an auxiliary because it appears to be the root of be.PST forms. However, nothing hinges on this label.
2. There is a further variant -er, which appears when the external argument is suppressed.

(11) a. ğur -es
die -PST.3PL
'They died.'

b. ğur **-ur** -an
die -PROG -PRS.3PL
'They are dying.'

(12) a. çit -es
chop -PST.3PL
'They chopped it.'

b. çit **-um** -an
chop -PROG -PRS.3PL
'They are chopping it.'

2.2 Allomorphy in Roots

I will be focusing on three roots that exhibit allomorphy: the Laz counterparts of 'eat', 'say', and 'go'. Among these, we find two cases of PROG portmanteau exponents, with 'say' and 'eat'. No distinct progressive suffix can follow these roots. Moreover, we find two cases of perfective stems that co-occur with the regular past tense suffixes.[3]. Table 1 presents their forms, along with a regular root 'break' for comparison. Various participle forms of all four roots are also provided for comparison in Table 2.

ASP/TNS	BREAK.3SG	EAT.3SG	SAY.3SG	GO.3SG
PRS PROG	t'ax-um-s	**imxor**-s	**it'ur**-s	mo-l-un
PAST PROG	t'ax-um-t'-u	**imxor**-t'-u	**it'ur**-t'-u	mo-l-ur-t'-u
PAST	t'ax-u	şk'om-u	**t'k'**-u	mo-**xt'**-u
OPTATIVE	t'ax-a-s	şk'om-a-s	**t'k'v**-a-s	mo-**xt'**-a-s

Table 1: Irregular progressive and perfective stems in Laz

	BREAK	EAT	SAY	GO
PRF PTCP	t'ax-eri	şk'om-eri	zit'-eri	mo-lv-eri
FUT PTCP	o-t'ax-oni	o-şk'om-oni	o-zit'-oni	mo-lv-oni
MASDAR	o-t'ax-u	o-şk'om-u	o-zit'-u	mo-lv-a

Table 2: Participle and masdar forms of irregular stems in Laz

Some notes regarding these forms are in order. First, I argue that the progressive stem *it'ur* is not decomposed as *it'+ur* where *-ur* is a regular progressive suffix. The reason for this is that the progressive suffix *-ur* always has a portmanteau *-un* for PROG.PRS.3SG as shown in (13), bleeding the insertion of the regular suffix *-s* for PRS.3SG shown in (14). Hence, we have the forms in (15), where *it'ur* corresponds to the portmanteau for say.PROG.

3. Table 1 also provides the optative forms. Optative marking is incompatible with progressive marking. Therefore, I take OPTATIVE to combine with perfective stems. I will not discuss optative forms in the rest of the paper.

(13) a. ğur -ur -an
die -PROG -3PL.PRS
'They are dying.'

b. ğur **-un**
die -PROG.3SG.PRS
'She/he is dying.'

(14) a. t'ax -um -an
break -PROG -3PL.PRS
'They are breaking it.'

b. t'ax -um -s
break -PROG -3SG.PRS
'She/he is breaking it.'

(15) a. it'ur -an
say.PROG -3PL.PRS
'They are saying it.'

b. it'ur **-s**
say.PROG -3SG.PRS
'She/he is saying it.'

Another minor point worth mentioning is that the 'elsewhere' form for 'go', i.e. *lv*, sometimes gets reduced to *l*. The same is true for *t'k'v* 'say' being reduced to *t'k'*. These reductions are due to a fully regular phonological process which reduces Cvu into Cu (Öztürk & Pöchtrager 2011).

Finally, there is a question about which root forms constitute the 'elsewhere' forms. This is a particularly pressing question for the root for 'say', given that it has distinct irregular stems for both progressive and perfective. In this regard, causativization seems informative in that it introduces a suffix between the root and the higher aspect related features, as shown in (16).

(16) a. o- t'ax **-ap** -am -s
CAUS break -CAUS -PROG -PRS.3SG
'They are making him break it.'

b. o- t'ax **-ap** -es
CAUS break -CAUS -PST.3PL
'They made him break it.'

Both 'eat' and 'say' have *-ap* causative forms. Progressive portmanteaus lose when *-ap* intervenes. Similarly, the perfective stem of 'say' loses when *-ap* intervenes. These facts are shown in (17) and (18).

(17) a.

	PROG portmanteau	'elsewhere'
EAT	**imxor**	şk'om

b. **imxor** -an
eat.PROG -PRS.3PL
'They are eating.'

c. o- şk'om **-ap** -es
CAUS eat -CAUS -PST.3PL
'They made him eat.'

d. o- şk'om -**ap** -am -an
CAUS eat -CAUS -PROG -PRS.3PL
'They are making him eat.' ⇐ portmanteau *imxor* loses

(18) a.

	PFV portmanteau	PROG portmanteau	'elsewhere'
SAY	**t'k'v**	**it'ur**	zit'

b. **tk'v** -es
say.PFV -PST.3PL
'They said.'

c. **it'ur** -an
say.PROG -PRS.3PL
'They are saying it.'

d. o- zit' -**ap** -es
CAUS say -CAUS -PST.3PL
'They made him say it.' ⇐ portmanteau *t'k'v* loses

e. o- zit' -**ap** -am -an
CAUS say -CAUS -PROG -PRS.3PL
'They are making him say it.' ⇐ portmanteau *i'tur* loses

Table 3 summarizes our findings.

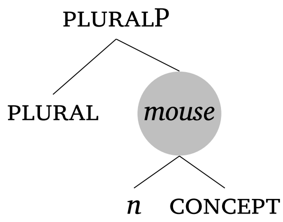
	PFV portmanteau	PROG portmanteau	'elsewhere'
EAT	n/a	**imxor**	şk'om
GO	**xt'**	n/a	lv
SAY	**t'k'v**	**it'ur**	zit'

Table 3: Root portmanteaus in Laz

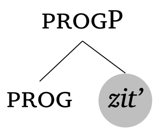
3 The baseline Nano analysis

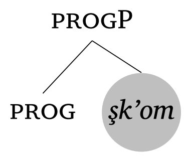
I assume phrasal spell-out (Caha 2009; Starke 2009), where each feature gets individually merged into the structure and triggers a cycle of lexicalisation. For ease of exposition, I follow the particular implementation in Caha, Clercq & Wyngaerd (2019) and Caha, De Clercq & Vanden Wyngaerd (2019). Accordingly, I assume that a root portmanteau is a lexically stored tree (LST) that contains a particular restriction on what got inserted in the previous cycle. To illustrate, the LST for (19) requires that the LST for *mouse* got inserted into the phrase built in the previous cycle.[4]

4. I thank Utku Türk for sharing the LaTeX code for drawing these diagrams that hopefully help the reader.

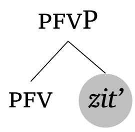
(19) *mice* ⟺ 

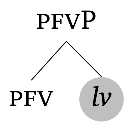
Accordingly, if we assume that the progressive corresponds to a single feature PROG, the LSTs for the progressive portmanteaus for 'say' and 'eat' will be as shown in (20) requiring that the LSTs for *zit'* and *şk'om* got inserted in the previous cycle, respectively.

(20) a. *it'ur* ⟺ 

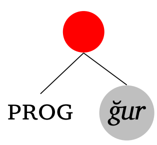
b. *imxor* ⟺ 

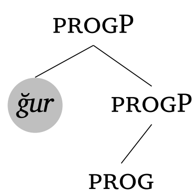
Recall that perfective stems for 'say' and 'go' also behave like portmanteaus in that they lose when the causative suffix intervenes. Hence, we can take them to be on par with progressive portmanteaus, as shown in (21).

(21) a. *t'k'v* ⟺ 

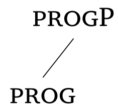
b. *xt'* ⟺ 

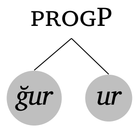
These LSTs demonstrate the easiest way to successfully lexicalise a given cycle. Of course, we do not have portmanteaus for everything. For example, consider the derivation in (22), which shows that the LST for the regular root *ǧur* 'die' got inserted in the previous cycle. But this time, the lexicon does not have a portmanteau LST that can be inserted into the root node, PROGP. When this is the case, the lexicalisation algorithm attempts certain rescue operations in a given order. In the case at hand, an evacuation movement moves out the complement of the newly merged feature PROG, i.e. the structure that the LST for *ǧur* got inserted into. This effectively makes *ǧur* a specifier of the phrase in this cycle and forms a new constituent PROGP.

(22) 

(23) 

Suppose (contrary to fact) that the Laz lexicon has the unique entry for PROGP, given in (24). Notably, this entry can be succesfully inserted into the newly formed PROGP in (23), completing the lexicalisation of this cycle, as shown in (25). Notice that neither of the progressive portmanteaus in (20) would be usable here, given that they come with specific requirements on what got inserted in the previous cycle.

(24) *-ur* ⟺ 

(25) 

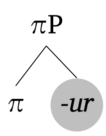
The instance of lexicalisation driven movement illustrated above, called *comp-to-spec* movement, in essence creates a suffix(al structure). The comp-to-spec movement is not the only type of lexicalisation driven movement available. As a matter of fact, the lexicalisation algorithm first tries evacuating the spec of the complement in a given cycle. If this movement fails to yield successful lexicalisation (or if the complement has no specifier), then comp-to-spec movement is attempted.

Let us now consider a case where the spec of the complement is moved out of the way, yielding a successful lexicalisation. Recall that the progressive marker *-ur* has a portmanteau *-un* for PRS.3SG, as shown in (26).

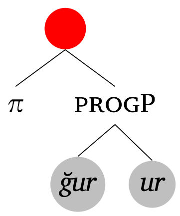
(26) a. ğur -ur -an
die -PROG -3PL.PRS
'They are dying.'

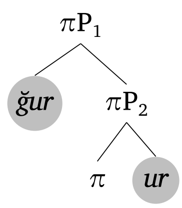
b. ğur **-un**
die -PROG.3SG.PRS
'She/he is dying.'

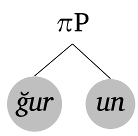
Suppose that this portmanteau *-un* corresponds to the lexical entry in (27), which comes with the restriction that the LST corresponding to *-ur* got inserted in the previous cycle.

(27) *-un* ⟺ 

Let us assume that in deriving (26b), the next feature to be merged in syntax is π introducing a person feature, as shown in (28), continuing the derivation in (25). Given that there is no portmanteau for the entire phrase πP, the first rescue movement in order, spec-to-spec movement, is attempted. Notice here that in the previous cycle, *ğur* was a complement that got moved and became a spec. Hence, the complement of the newly merged feature π has a spec that we can apply spec-to-spec movement to. This results in the structure in (29). Notably, this spec-to-spec movement creates the right context for the insertion of the LST for *-un*, leading to the successful lexicalisation of this cycle, as shown in (30). It should be noted that the presence of the portmanteau *-un* in the lexicon, which is able to override the progressive suffix *-ur*, allowed a successful lexicalisation here in the first attempt. If the lexicon did not have this portmanteau, spec-to-spec movement would fail to provide a successful lexicalisation, and comp-to-spec movement would be attempted, creating a suffixal structure.

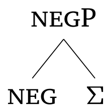
(28)


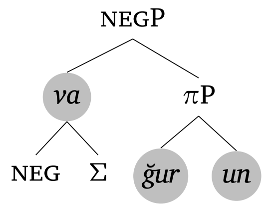
(29)


(30)


So far, we have seen how a new suffixal structure is created by comp-to-spec movement and how a suffix may override another suffix via spec-to-spec movement. In some cases, neither option will yield a successful lexicalisation, leading to a truly last resort operation: spawining a new derivation and building a complex left branch (CLB) and merging it in the main derivational spine. This last resort gives us prefixes. As an example, let us consider the negation prefix in Laz, *va-*. Suppose further that the LST for *va-* is as in (32). An important thing to notice about this LST is that unlike a suffix, it is binary branching. There is no movement in syntax that would create this sort of constituent. Supposing that the only possible candidate LST for lexicalising the NEG feature, is the one in (32), the derivation is forced into spawning a new derivation and building this CLB and merging it with the main derivational spine, as shown in (33). What is crucial here is that when the CLB built in a separate derivational workspace is merged into the main derivation, it projects, introducing the feature NEG into the main derivation. In that sense, it is a (complex) head, rather than a specifier. Accordingly, it is not eligible for spec-to-spec movement.

(31) va- ğur **-un**
NEG- die -PROG.3SG.PRS
'She/he is not dying.'

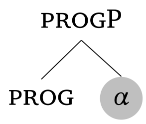
(32) *va-* $\Longleftrightarrow$ 

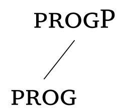
(33) 

4 What can and cannot block root portmanteaus?

In the previous section, I have fleshed out a baseline Nano analysis of the root portmanteaus in Laz while also demonstrating the adopted lexicalisation algorithm. In this section, I will discuss various configurations with regard to their potential to render root portmanteaus unusable, focusing on the behaviour of prefixes. Before we start discussing the actual data, let us first consider some hypothetical scenarios in which a root portmanteau may be blocked. Let β be a portmanteau exponent which normally bleeds the insertion of a regular PROG affix ϕ as well as 'elsewhere' form of the root α. In other words, we expect β to bleed the bimorphemic lexicalisation in (34b). The relevant hypothetical LSTs are as in (35).

(34) a. β
b. *α-ϕ

(35) a. $\beta \Longleftrightarrow$ 

b. $\phi \Longleftrightarrow$ 

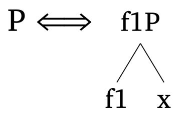
The lexicalisation algorithm laid out in the previous section predicts that a feature that is below PROG realized as a suffix S will prevent β from appearing when it is part of the structure, as shown in (36b). Rather, we expect the lexicalisation in (36b), where α and ϕ appear. Blocking patterns of this kind are well-attested and are widely discussed in the literature. Therefore, I will not dwell on it further here.

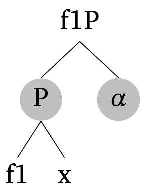
(36) a. *β-S
b. α-S-ϕ

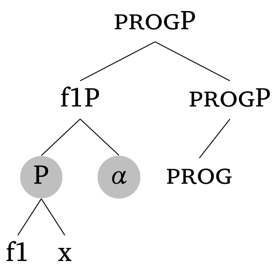
What is more intriguing is the case in (37a), where a prefix that lexicalises a feature that is below PROG blocks the portmanteau β, forcing the bimorphemic lexicalisation in (37b) where α and ϕ appear adjacent, a string that is otherwise bled by β. If such cases do exist, then we have robust evidence for structural intervention. This is because although P is a prefix, it still *non-linearly* bleeds the insertion of β.

(37) a. *P-β
b. P-α-ϕ

Does Nano predict this state of affairs? The answer is affirmative. Provided that P is a **projecting CLB** and furthermore is lexicalising a feature below PROG, it will always stay in the way of β, blocking its insertion. This is schematized in (38). Suppose the prefix P is the only way to lexicalise the feature f1 (along with some other feature x), using the LST in (38a). This means that the prefix is a projecting CLB as shown in (38b). As a result, when the derivation proceeds into the next cycle and merges PROG, spec-to-spec movement cannot evacuate P, giving way to the next available evacuation movement, i.e. comp-to-spec movement, yielding the structure in (38c) and the lexicalisation in (37b) using the regular suffix ϕ in (35b).

(38) a.


b.


c.


In Laz, we do find non-linear blocking effects of the kind illustrated above. In other words, linearly uninvolved morphemes can block root portmanteaus. But the more important question is: which ones can? Do we make the right predictions? I argue that we do.

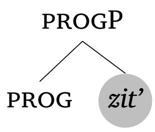
There are four preverbal 'slots' in Laz, as illustrated in (39). I will show that among these four group of prefixes, it is only PRV elements that can block root portmanteaus, while polarity, agreement, and spatial markers do not block root portmanteaus. I will argue that polarity, agreement prefixes expone features higher in the structure than any feature that realize the root portmanteaus, hence are irrelevant. I will argue that spatial markers end up as prefixes as a result of spec-to-spec movement, i.e. they are not projecting CLBs. Finally, I will show that PRV elements all expone features that structurally intervene for the insertion of the root portmanteaus.

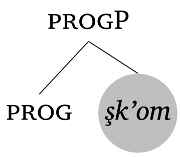
(39) a. POLARITY + SPATIAL + AGREEMENT + PRV + $\sqrt{\ }$ + ...

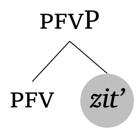
b. va- ce- v- o- çum -ap -i
NEG DOWN 1 CAUS HIT -CAUS -PST
'I didn't let them beat him.'

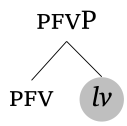
4.1 Higher prefixes do not block portmanteaus

Recall that I have posited the LSTs below for root portmanteaus, which all override the insertion in the previous cycle to realize an aspect feature: PROG or PFV.

(40) a. *it'ur* ⟺ 

b. *imxor* ⟺ 

(41) a. *t'k'v* ⟺ 

b. *xt'* ⟺ 

In what follows, I discuss polarity and agreement prefixes to show that they expone features higher than PFV and PROG, hence are unable to block these portmanteaus.

4.1.1 Polarity prefixes

Polarity markers NEG and AFF do not block the root portmanteaus in Laz, as shown in (42). I argue that this state of affairs is predicted in that these prefixes expone features higher than aspect features (hence come much later in the derivation).

(42) a. do- t'k'v -es
AFF- say.PFV -PST.3PL
'They *did* say.' elsewhere *zit'*

b. var- imxor -an
NEG- eat.PROG -PRS.3PL
'They are not eating.' elsewhere *şk'om*

c. va- mo- xt' -es
NEG- TWRD-SPKR- go.PFV -PST.3PL
'They did not come.' elsewhere *lv*

The only thing that can precede polarity markers is a run-of-the-mill subordinator *na-*, as shown in (43). Since Laz is mostly suffixing on the verb, this alone does not help us out.

(43) [Dişk'a **na-** var- çit -u] t'k'-u
wood COMP- NEG- chop -PST.3SG say.PFV-PST.3SG
'He said he didn't chop wood.'

However, there is some informative variation in the realization of NEG contingent on mood. While the indicative negation is *va(r)-*, in the irrealis mood we find what appears to be an additional prefix *ti-* following it, giving us *va + ti-* as a complex form of negation, as shown in (44b). Notably, we can deduce where *ti-* may be located in the structure. The irrealis is also marked suffixally, and crucially as a suffix that follows tense suffixes.

(44) a. va- t'ax -i
NEG- break -PST.2SG
'You didn't break it.'

b. va + ti- t'ax -i -k'o
NEG + IRR- break -PST.2SG -IRR
'You wouldn't break it'
'You weren't gonna break it.'

Given that NEG has a possibly decomposable irrealis form and that IRR features are apparently above tense, it seems safe to assume that features that give us the polarity prefixes are at least above the aspect features in the fseq. If this is correct, then polarity affixes have no way of blocking PROPP and PFVP portmanteaus.

4.1.2 Agreement prefixes

Prefixal agreement markers do not block portmanteaus, as shown in (45).

(45) a. v- imxor
1 eat.PROG
'I am eating.'

b. p'- t'k'v -i -t
1 say.PFV -PST -
'We said.'

The agreement system of South Caucasian is notoriously complex, having both suffixal and prefixal agreement for both subjects and objects.

There is also an intriguing interplay between suffixal and prefixal agreement. I will follow Blix (2021) who argues that (person and number) agreement features are just below the tense features in the fseq. There is clear evidence that suffixal agreement is located around the tense region, for tense affixes are portmanteaus for tense and agreement. For example, *-es* realizes PST.3.PL. Importantly, there is evidence that suffixal and prefixal agreement together lexicalise a contiguous region in the fseq, which allows us to locate agreement prefixes around the tense region, as well.

In support of this, I will mention an additional portmanteau.[5] While *v-/p-* is a first person subject marker that shows up in the absence of a second person object, its insertion is blocked by an additional root allomorph *ft'* just in case the PFVP portmanteau *xt'* got inserted in the previous cycle. This is summarized in (46). The relevant blocking effect is shown in (47).

(46) a. lv → go
b. xt' → go + PFV
c. ft' → go + PFV + 1

(47) a. gama- **ft'** -i
OUT go.PFV.1 -PST
'I went out.'
b. *gama- p- **xt'** -i
OUT 1- go.PFV -PST
'I went out.'

This informs us that the insertion of ft' is contingent on the successful insertion of *xt'* (i.e. a PFVP), locating the prefixal agreement markers above aspect features. If so, we correctly predict that prefixal agreement has no way of blocking aspect portmanteaus containing a root.

(48) *ft'* ⟺ ['first person agreement features' *xt'*]

4.2 Prefixes that give way to portmanteaus

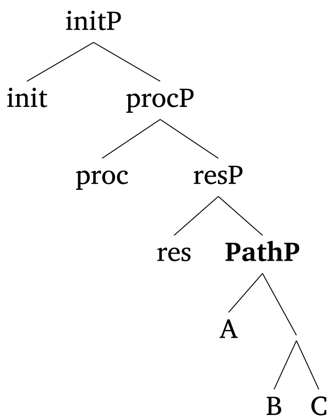
Spatial markers, which form a large set consisting of simplex and complex forms, do not block portmanteaus, as shown in (49).

5. To my knowledge, pxt' → ft' is not a regular phonological process in Laz. But this may turn out to be wrong.

(49) a. oxori-şe gama- xt' -es
house-ABL OUT go.PFV -PST.3PL
'They went out of the house.' elsewhere *lv*

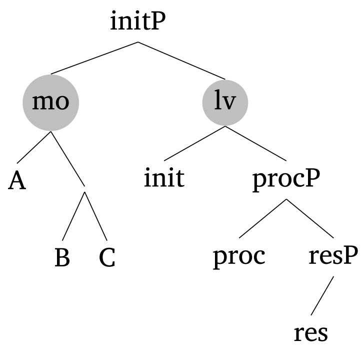
b. livadi-s do + lo- xt' -es
garden-DAT INTO.DOWN- go.PFV -PRS.3PL
'They went down into the garden.'

I follow Starke (2018) in assuming that spatial markers lexicalise a PathP at the very bottom of the fseq.[6] Assuming that verbal features are the *init, proc, res* of Ramchand (2008), this gives us representations like in (50).

(50)


Following Starke (2018), I assume that the PathP is moved to the left of the verb via comp-to-spec movement in order to create a constituent for inserting the root. An illustration is in (51).

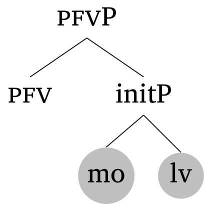
(51) a. mo- l[v] -un
TWRD.SPEAKER- go -PROG.PRS.3SG
'She/he is coming.'

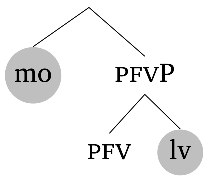
b.


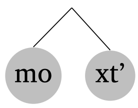
6. See also Eren (2016) on spatial markers in Laz.

An important feature of spatial prefixes is that they start out as complements in the main derivation and end up as specifiers as a result of comp-to-spec movement. Therefore, we predict that they can give way to root portmanteaus via spec-to-spec movement. This is illustrated in (52), where the PFVP portmanteau *xt’* overrides *lv* when the PathP *mo-* gets spec-to-spec moved out of the way. Notably, this is possible precisely because it has been comp-to-spec moved into that position. In other words, it is not an immobile projecting CLB (Starke 2018; Caha 2019).

(52) a. mo- xt’ -u
TWRD.SPEAKER- go -PST.3SG
‘She/he came.’

b. 

c. 

d. 

One question that remains unresolved in this line of thinking is the order between the prefixes. If agreement features are in the tense region in the fseq, we do not expect spatial prefixes to precede prefixes that expone those features. This is because for a complex left branch to be built, all evacuation movements must have failed, predicting that the spatial prefixes should be trapped below along with the root and lower affixes. I leave this as a genuine puzzle for future research.

4.3 Projecting CLBs block portmanteaus

We are left with pre-root vowels in the prefixal domain. PRV is a slot that can host one of these vowels: {o, u, i, a} (Demirok 2011; 2013; Öztürk 2013; Taylan & Öztürk 2014). o- occurs with an additional overt suffix. So, I focus on {u, i, a} here.

(53) t’ax -u
break -PST.3SG
‘She broke it.’

(54) **o-** t’ax **-ap** -u
CAUS- break -CAUS -PST.3SG
‘She made him break it.’ causative

(55) **u**- t'ax -u
APPL- break -PST.3SG
'She broke it for him.' applicative (nonreflexive)

(56) **i**- t'ax -u
PASS/REFL- break -PST.3SG
'(Someone) broke it.' impersonal passive
'She broke it for herself.' reflexive-applicative

(57) **a**- t'ax -u
ABIL- break -PST.3SG
'She was able to break it.' root modality/ability

The crucial observation is that unlike all other prefixes, pre-root vowels systematically block the root portmanteaus.

The eat.PROG portmanteau *imxor* is blocked by the PRV *u-*, leading to the bimorphemic lexicalisation with the elsewhere form of the root and a regular progressive suffix.

(58) a. imxor -an
eat.PROG -PRS.3PL
'They are eating.'

b. **u-** şk'om -am -an
APPL- eat -PROG -PRS.3PL
'They are eating something that belongs to him.'

The go.PFV portmanteau *xt'* is blocked by the PRV *a-*, as shown in (59).

(59) a. mo- xt' -u
TWRD-SPKR- go.PFV -PST.3SG
'She came.'

b. mv- **a-** l[v] -u
TWRD-SPKR- ABILITY go -PST.3SG
'She was able to come.'

c. *mv-a-xt'-u

Both say.PFV and say.PROG portmanteaus are blocked by all PRVs, as shown in (60) and (61).

(60) a. **t'k'** -u
say.PFV -PST.3SG
'She said it.'

b. **i-** zit' -u
PASS- say -PST.3SG
'(Someone) said it.'

c. **a-** zit' -u
ABIL- say -PST.3SG
'She was able to say it.'

d. **u-** zit' -u
APPL- say -PST.3SG
'She said something about him.'

e. *i-t'k'-u, *a-t'k'-u, *u-t'k'-u

(61) a. **it'ur** -an
say.PROG -PRS.3PL
'They are saying it.'

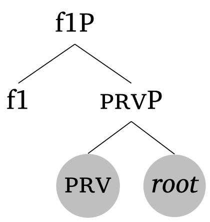
b. **i-** zit' -en
PASS- say -PROG.PRS.3SG
'It is said.'

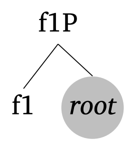
c. **a-** zit' -er -an
ABIL- say -PROG -PRS.3PL
'She is able to say it.'

d. **u-** zit' -am -an
APPL- say -PST.3PL
'They are saying it about him.'

PRVs require an in-depth investigation. But I think it is fairly reasonable to say that they are lexicalising features that are below aspect features. In familiar logical forms, causative, passive, applicative and root modal projections, in particular ability modals, compose with event predicates (Hacquard 2006; Pylkkänen 2002; Demirok 2018). If this is the case, they must be lower in the fseq than aspect and tense nodes. Accordingly, I assume that they all lexicalise complex left branches in the 'VP zone', and are lower in the fseq than aspect features.

As schematically shown in (62), when a PRV is built in a newly spawned workspace and is merged into the main derivation, it will project. As a result, when the derivation proceeds to the next cycle merging some feature f1, PRV cannot be subject to the spec-to-spec movement, bleeding the insertion of a possible portmanteau β like the one in (63).

(62)


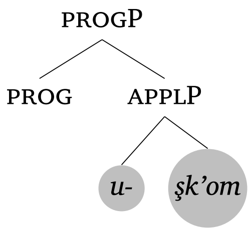
(63) $\beta \Longleftrightarrow$


Let us finalize our discussion by seeing how a PROGP portmanteau is blocked on a real example. Recall the bimorphemic lexicalisation forced by the APPL prefix in (64). All evacuation movements have failed and the derivation, as a last resort, has built the prefix *u-* to lexicalise the APPL feature and has merged it as a projecting CLB, as shown in (65). In the next cycle when PROG is merged, spec-to-spec movement will fail to move this projecting CLB, i.e. u-, which renders the LST in (66) unusable.

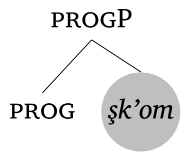
(64) a. imxor -an
eat.PROG -PRS.3PL
'They are eating.'

b. **u-** şk'om -am -an
APPL- eat -PROG -PRS.3PL
'They are eating for him/something that belongs to him.'

(65)

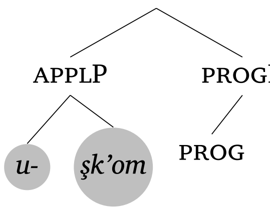


(66) *imxor* ⟺

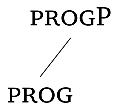


The derivation proceeds with attempting a comp-to-spec movement, which gives us the structure in (67), allowing the PROG suffix in (68) to be inserted in the right branch.

(67)



(68) *-am* ⟺



5 Concluding Remarks

If what I have been saying about Laz is on the right track, we have evidence that the lexicalisation algorithm in Nano makes the correct predictions about when a prefix may or may not block a root portmanteau. I have argued that prefixes in Laz fall in three classes. First, polarity and agreement prefixes are correctly predicted to not block the root portmanteaus, for they seem to be exponing features higher than features that the portmanteaus lexicalise. Second, spatial prefixes are also correctly predicted not to block portmanteaus if they are specifiers that originate very low in the structure but get comp-to-spec moved to give way to the root insertion. Finally, I have tried to show that PRVs in Laz are projecting complex left branches and crucially expone features that are in the way of the portmanteau, i.e. lower than aspect features. Given their projecting nature, they cannot be moved out, rendering any portmanteau that reaches up to higher features unusable.

This brief investigation has surely glossed over many details such as the allomorphy among progressive suffixes and the inner composition of the PRVs and how they interact. I leave a fuller investigation to a future occasion.

References

Blix, Hagen. 2021. Spans in South Caucasian agreement. *Natural Language & Linguistic Theory* 39. 1–55. https://doi.org/10.1007/s11049-020-09475-x.

Bobaljik, Jonathan D. 2012. *Universals in Comparative Morphology: Suppletion, superlatives and the structure of words*. Cambridge, MA: MIT Press.

Božič, Jurij. 2018. Generalizations on Root Suppletion: Motivating a Theory of Contextual Allomorphy. In *Proceedings of the 35th West Coast Conference on Formal Linguistics*, 114–123. Somerville, MA: Cascadilla Proceedings Project.

Caha, Pavel. 2009. *The Nanosyntax of Case*. University of Tromsø dissertation.

Caha, Pavel. 2018. Notes on insertion in Distributed Morphology and Nanosyntax). In Lena Baunaz, Karen De Clercq, Liliane Haegeman & Eric Lander (eds.), *Exploring Nanosyntax*, 57–87. Oxford University Press, Oxford.

Caha, Pavel. 2019. *Case competition in Nanosyntax. A study of numerals in Ossetic and Russian*. Ms. Masaryk U.

Caha, Pavel, Karen De Clercq & Guido Vanden Wyngaerd. 2019. *On the difference between a $\sqrt{\ }$ and a root.* Ms, available at https://ling.auf.net/lingbuzz/004391.

Caha, Pavel, Karen De Clercq & Guido Vanden Wyngaerd. 2019. The Fine Structure of the Comparative. *Studia Linguistica* 73(3). 470–521.

Choi, Jaehoon & Heidi Harley. 2019. Locality domains and morphological rules. *Natural Language & Linguistic Theory* 37(4). 1319–1365.

Demirok, Ömer. 2011. The Verb. In Balkız Öztürk & Markus Pöchtrager (eds.), *Pazar Laz*. LINCOM.

Demirok, Ömer. 2013. AGREE as a Unidirectional Operation: Evidence from Laz. Unpublished MA Thesis., Bogazici University.

Demirok, Ömer. 2018. A Modal Approach to Dative Subjects in Laz. In *Proceedings of NELS 48*.

Embick, David. 2010. *Localism versus globalism in morphology and phonology*. Cambridge, MA: MIT Press.

Eren, Ömer. 2016. *Spatial prefixes of Pazar Laz: a nano-syntactic approach.* MA thesis, Boğaziçi University.

Gouskova, Maria & Jonathan David Bobaljik. 2019. Allomorphy and Vocabulary Insertion. Ms, NYU & Harvard.

Hacquard, Valentine. 2006. *Aspects of Modality*. MIT dissertation.

Halle, Morris & Alec Marantz. 1993. Distributed Morphology and the Pieces of Inflection. In *In The View from Building 20, MIT Working Papers in Linguistics*, 111–176. Cambridge, MA: MIT Press.

Merchant, Jason. 2015. How much context is enough? Two cases of span-conditioned stem allomorphy. *Linguistic Inquiry* 46. 273–303. `https://doi.org/10.1162/LING_a_00182`.

Ostrove, Jason. 2018. Stretching, spanning, and linear adjacency in Vocabulary Insertion. *Natural Language & Linguistic Theory* 36(4). 1263–1289. `https://doi.org/10.1007/s11049-018-9399-y`.

Öztürk, Balkız. 2013. Low, high and higher applicatives. In Victoria Camacho-Taboada, Ángel L. Jiménez Fernández, Javier Martín-González & Mariano Reyes-Tejedor (eds.), *Linguistik Aktuell/Linguistics Today 197*, 275–296. John Benjamins.

Öztürk, Balkız & Markus Pöchtrager. 2011. *Pazar Laz*. LINCOM.

Pylkkänen, Liina. 2002. *Introducing Arguments*. MIT dissertation.

Ramchand, Gillian. 2008. *Verb Meaning and the Lexicon: A First Phase Syntax*. Cambridge University Press.

Smith, Peter W., Beata Moskal, Ting Xu, Jungmin Kang & Jonathan David Bobaljik. 2019. Case and number suppletion in pronouns. *Natural Language & Linguistic Theory* 37. 1029–1101.

Starke, Michal. 2009. Nanosyntax: A short primer to a new approach to language. *Nordlyd* 36. 1–6.

Starke, Michal. 2018. Complex left branches, spellout, and prefixes. In Lena Baunaz, Karen De Clercq, Liliane Haegeman & Eric Lander (eds.), *Exploring Nanosyntax*. Oxford: Oxford University Press.

Taylan, Eser Erguvanlı & Balkız Öztürk. 2014. Transitivity in Pazar Laz. *Acta Linguistica Hungarica* (61). 271–296.