Case stacking in nanosyntax

Anke Assmann[*]
Leipzig University

2014

In this paper, I discuss a possibility for extending the nanosyntactic approach to case assignment and realization in order to model case stacking. Building on the work by Caha (2009), I show that nanosyntax can derive instances of overt case stacking as well as abstract case stacking. The main idea is that a case checking head can attract more than one K(ase)P. This leads to configurations where more than one K-head in the case sequence of one argument is checked. Overt case stacking comes about if all checked cases in one case sequence are realized separately. Abstract case stacking results if only one of the checked KPs is realized.

1 Introduction

In this paper, I discuss a possibility for extending the nanosyntactic approach to case assignment and realization in order to model case stacking. The term *case stacking* refers to structures, where a DP is marked for more than one case (McCreight 1988; Nordlinger 1998; Merchant 2006; Richards 2013; Pesetsky 2013). This usually occurs in configurations where a DP is embedded in another DP, for example in possessive constructions, which are the main empirical focus of this paper. In addition to its own possessive case marker, the possessor bears the case of the entire possessive DP. Instances of case stacking are also referred to as *suffixaufnahme* (Plank 1995), a term which also subsumes stacking of agreement markers. An example for overt case stacking is given in (1) from Huallaga Quechua.

(1) hipash-nin-ta kuya-: Hwan-pa-ta
daughter-3POSS-ACC love-1 Juan-GEN-ACC
'I love Juan's daughter.' (Plank 1995, 47)

In (1), the possessor *Juan* bears both the genitive case marker *-pa* and the accusative case marker, which is the case of the entire possessive DP.[1].

[*]I would like to thank Doreen Georgi, Timo Klein, Sandhya Sundaresan, and Philipp Weisser for helpful comments and discussion. This research is supported by DFG-grant MU 144/8-1 ("Local modeling of non-local dependencies in syntax").

[1]Note that the possessive DP in (1) is split. The possessum *daughter* is topicalized. I come back to this split in section 3.2.1

Additionally to overt case stacking, Pesetsky (2013) and Assmann et al. (2014) argue for Russian and Udmurt respectively that case stacking might also apply abstractly: the DP bears two abstract cases, but only one case marker shows up overtly. Pesetsky (2013) uses abstract case stacking to explain the case mismatches in Russian paucal numerals. Assmann et al. (2014) show that abstract case stacking elegantly derives the case split on possessors in Udmurt.

In general, case stacking raises two theoretical question. First, a special assumption about case assignment must be made since a DP receives only one case. However, the existence of case stacking phenomena implies that a case bearing category must, in principle, be able to receive more than one case. The question is what kind of mechanism allows multiple case assignment. Second, for cases of abstract case stacking, the concrete overt case marker must be determined. The question is which principles determine the choice?

In the minimalist program, these questions can be answered as follows: As for the first question, since case assignment is standardly assumed to be an instance of Agree, an additional Agree relation besides the standard case assignment relations is needed. This could either be case concord between the possessum and the possessor (e.g. downward spreading, see Matushansky 2008; Bjorkman 2013; Erlewine 2013) or a direct relation between the possessor and the head that assigns case to the possessum (e.g. Multiple Agree, see Hiraiwa 2001; Vainikka and Brattico 2014).

As for the second question, the choice for the only overt marker in abstract case stacking configurations is not determined in narrow syntax. Instead the several case feature values the possessor has received in syntax, are once again manipulated in a postsyntactic morphological component before the case features are realized by markers. This manipulation could involve simple deletion of all but one feature or the computation of a completely new case feature (see Assmann et al. 2014 for such an approach.)

But how can the two problems of case stacking be solved in a nanosyntactic framework? The first problem is that case assignment is modeled as syntactic movement. The even bigger problem is that nanosyntax does not assume a separate level of morphology. Rather, every morphosyntactic feature is a single syntactic head. Building on the work by Caha (2009), I show that nanosyntax can derive instances of overt case stacking as well as abstract case stacking. The main idea is that a case checking head can attract more than one K(ase)P. This leads to configurations where more than one K-head in the case sequence of one argument is checked. Overt case stacking comes about if all checked cases in one case sequence are realized separately. Abstract case stacking results if only the KP that has been stranded by additional case movement is realized.

The paper is structured as follows: Section 2 will summarize the relevant assumptions about case assignment in nanosyntax as presented in Caha (2009). In section 3, an extension to the framework is suggested that seems to be necessary in order to derive instances of overt case stacking. In section 4, the analysis is applied to an instance of abstract case stacking in Udmurt. Section 5 concludes.

2 Case assignment in nanosyntax

In nanosyntax, every category that constitutes a feature in standard minimalism is a separate syntactic head (an extension of cartographic approaches to syntax, see e.g. Cinque 2002; Rizzi 2004; Belletti 2004). These heads are merged in syntax and result in complex morphosyntactic structures that are translated into phonological and semantic representations with the help of a lexicon in which all translation rules are stored. This spell-out of syntactic structure proceeds cyclically and bottom-up.

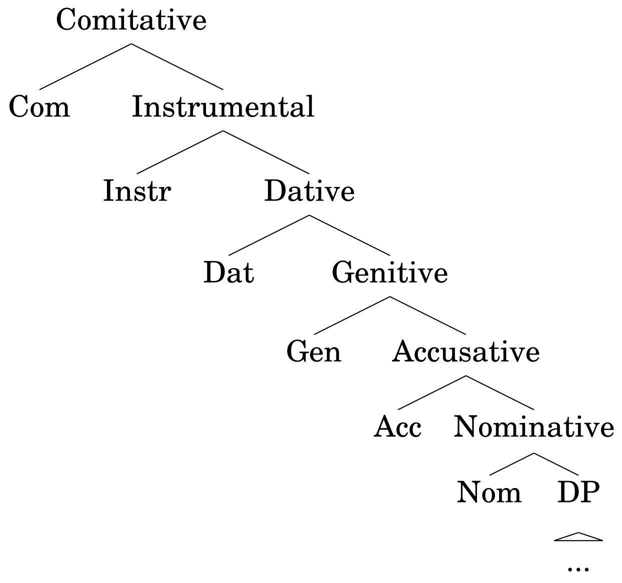
One of the main consequences of these assumptions is that there is no independent morphological component that can, additionally to the syntactic component, manipulate features the way it is done in e.g. Distributed Morphology. Rather, the syntactic operations of Merge and Move are the only tools we have in order to derive a morphosyntactic surface structure.

As for case, the nanosyntactic approach assumes that every case is represented by a head in syntax, which is merged above the DP layer of arguments. Moreover, these case heads are ordered on a functional sequence which corresponds to the typologically well-established case hierarchy of syncretisms in (2) (cf. Baerman et al. 2005).

(2) Nominative/Absolutive > Accusative/Ergative > Oblique cases

Under a nanosyntactic view, every head X that corresponds to a case C_1 is dominated by a head Y that corresponds to a case C_2 directly below C_1 on the hierarchy. The explicit universal case sequence proposed by Caha (2009) is given in (3). A case high on the case hierarchy in (2) corresponds to a case head low in the functional sequence.

(3)

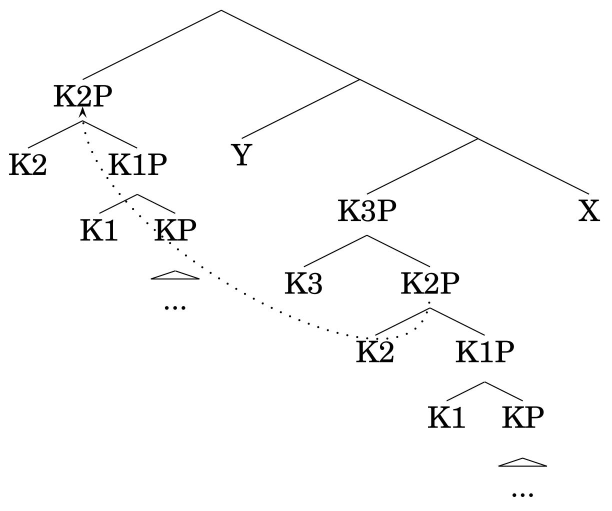


In syntax, every DP is generated with a number of case layers on top of it. The resulting phrases, henceforth KPs, are merged in argument positions whereby the case features on top of a DP must be appropriate for the θ-role associated with this position. Thus, instruments are always generated with the instrumental, recipients bear dative, themes accusative and so on.

Within the KP, the DP can move into any KP-layer. The KP-layers which are lower than the moved DP are realized as suffixes or postpositions. Higher layers are realized as prefixes or prepositions.

Certain case features of KPs must be checked in the syntax against special case checking heads. Case checking is done by movement of a case layer to the specifier of the respective case checking head. This results in stranding of higher case layers. Therefore, the case checking theory in nanosyntax is also referred to as the "Peeling Theory" of case assignment. The abstract schema is given in (4).

(4)



In (4), K3P is first merged into its θ-position as the sister of a head X. Afterwards for case reasons it has to move to the specifier of another head Y. However, since Y only checks case K2, the K3-layer is stranded in its base position.

After syntax, the trees are interpreted phonologically and semantically. Note that, as regards the spell-out of case markers, both the checked KPs as well as the stranded peels can in principle be realized (Caha, 2009, 157ff). Morphological realization is governed by the four principles given in (5) to (8).

(5) *The Superset Principle* (Starke 2005, Caha 2009, 55):
A phonological exponent is inserted into a node if its lexical entry has a (sub-)constituent which matches that node.

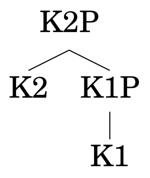
(6) *Match* (Caha 2009, 67):
A lexical constituent matches a node in the syntax if it is identical to that node, ignoring traces and spelled out constituents.

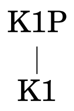
(7) *The Elsewhere Condition* (Caha 2009, 55):
In case two rules, R1 and R2, can apply in an environment E, R1 takes precedence over R2 if it applies in a proper subset of environments compared to R2.

(8) *The Anchor Condition* (Caha 2009, 89):
In a lexical entry, the feature which is lowest in the functional sequence must be matched against the syntactic structure.

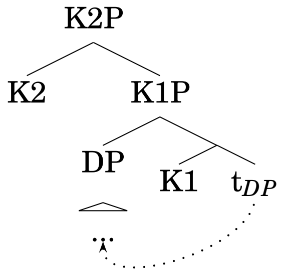
The way these four principles work together is illustrated by the abstract example in (9)–(10). (9) shows two hypothetical examples of lexical entries for case markers. (10) shows a structure created in syntax.

(9) *Lexical Entries*

a. /ba-/ ⇔ 

b. /-ab/ ⇔ 

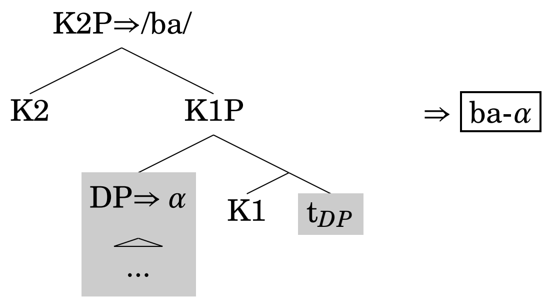
(10) *Syntactic Structure 1*



First, since only /-ab/ is a suffix, but /ba-/ is a prefix, the DP in (10) must to the specifier of K1 (otherwise the structure could not be spelled-out assuming that lexical entries are sensitive to the prefix-suffix distinction). In principle, a DP can move above every case head that licenses realization as a suffix (Caha 2009, 53).

Next, the case features must be realized. Following the Superset Principle in (5), the lexical entry must be identical to or be a superset of the features in the syntactic structure. Since both K1 and K2 should be realized, only the rule in (9a) can apply.[2] The marker /ba-/ matches the structure because traces and spelled-out constituents (shown in gray boxes in (11)) can be ignored according to Match in (6), that is, the moved DP and its trace are invisible for the spell-out rules for the case features.

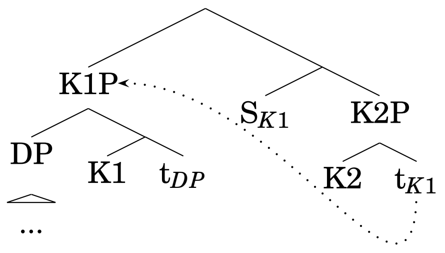
(11)



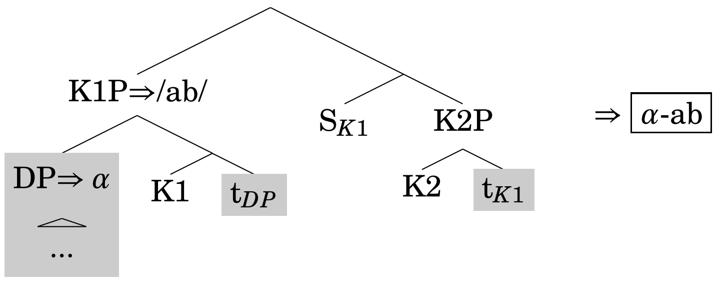
Next, assume that the derivation proceeds slightly differently: instead of checking the higher case K2, K1 is checked in the syntax. Then, K1P in (11) has to move to a case-checking head. The structure is given in (12).

(12) *Syntactic Structure 2*

[2] Caha (2009) discusses cases of compound case markers. He assumes that first a lower KP is spelled out and afterwards a higher KP is realized. This raises the question as to what the contexts for compound case marking are and in which contexts a higher KP must be realized even if there is a lexical entry for a lower KP. I will not discuss this issue here.



The moved K1P can be realized by applying either (9a) or (9b). Both markers constitute supersets of K1P in (12). However, due to the Elsewhere Condition (7), only marker /-ab/ can be inserted, since it can only be used in one environment, while the rule inserting /ba-/ can apply in two different environments ((11) and (12)). Thus, K1P in (12) is realized by /-ab/. Turning to K2P, none of the two markers can be used because /-ab/ is not a match and /ba-/ – despite constituting a superset of K2P in (12) under certain assumptions – is not applicable due to the Anchor Condition (8), which demands that a feature K1 has to be in the structure realized by /ba-/. Consequently, K2P does not receive a morphological exponent. Note, however, that stranded peels can in principle be spelled out (Caha 2009, 157ff.).

(13)


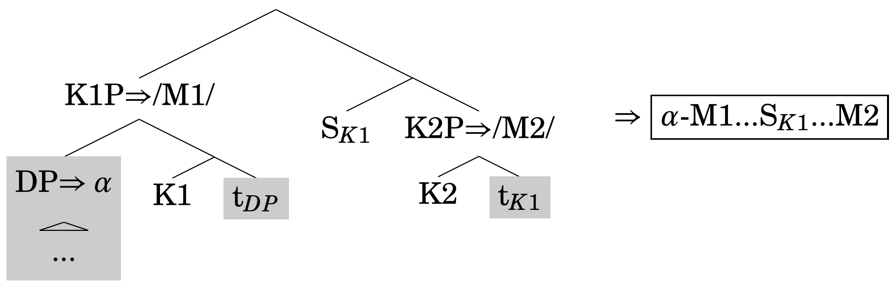
This concludes the summary of Caha's approach to case assignment. We can now turn to the analysis of the case stacking.

3 Case stacking in nanosyntax

The peeling theory of case assignment is not designed to account for instances of case stacking in the world's languages. Rather, as the system stands now, it rules out any instance of case stacking. The reason is that the different structures which can potentially be spelled out as case markers are separated by other material due to case movement and there are no morphological rules which can reunite them. To illustrate this problem, assume that both the moved K1P and the stranded K2P in (13) could be spelled out by some markers M1 and M2.[3] But since the two markers are disconnected in the structure due to movement, the surface structure does not point to a case stacking configuration. The structure is given again in (14). As we see the two case markers M1 and M2 are disconnected by (possibly overt) material in between K1P and K2P.

[3]In fact, this is the analysis for applicative marking in Mokilese (Oceanic) and Chichewa (Bantu) in Caha (2009, 157ff.).

(14)

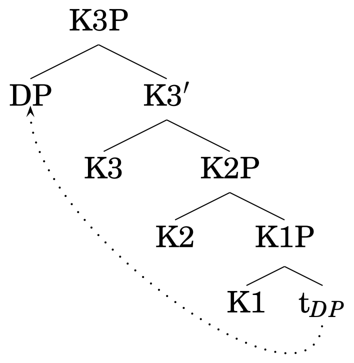


In this section, I will suggest some extensions and adjustments to the nanosyntactic framework to rule in case stacking. Ultimately, the analysis of case stacking will be the same as for applicative markers, the only difference being that, in the former, the two markers occur in the same phrase due to pied-piping of K2P when K1P moves.

3.1 Assumptions

The first assumption concerns DP movement within the KP. Since virtually all case stacking languages have only suffixal case markers, I assume (following Caha's approach) that the DP in case stacking languages has to move up to the highest case layer. Then, all case markers will follow the DP.

(15)

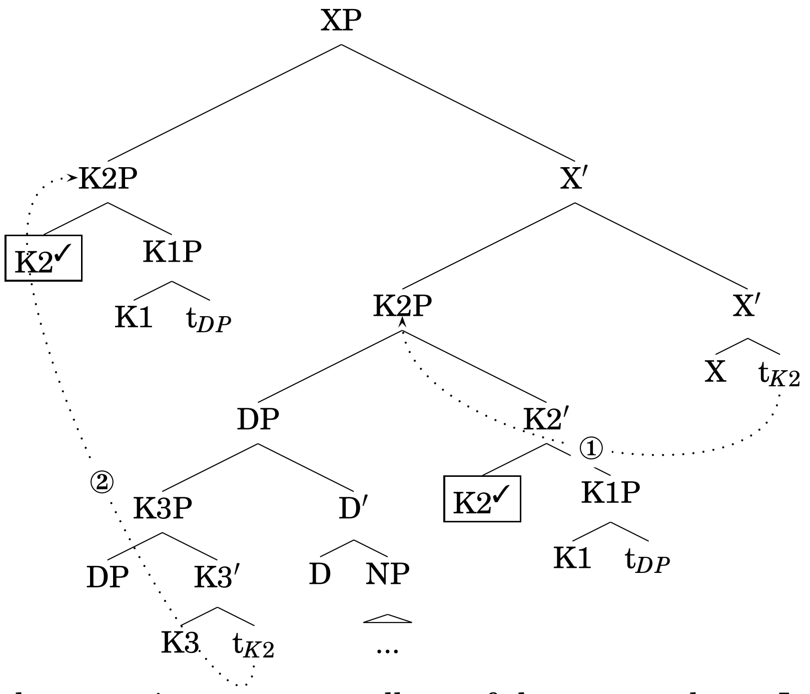


The second assumption concerns multiple case checking. Since there is no separate morphological component in nanosyntax, case stacking cannot be a postsyntactic phenomenon. Rather, it has to come about by multiple case checking. In order for this to occur, a case checking head X must be able to attract more than one KP.[4] More concretely, once X has attracted a KP to its specifier, it must be able to attract a KP of an argument embedded in its specifier.[5] The abstract schema is given in (16).

[4]An alternative would be to assume that there are multiple case checking heads X (Sandhya Sundaresan, p.c.), each of which checks one KP. This solution is discussed in section 3.3, where it is shown that an analysis involving multiple case heads needs additional assumptions to predict case stacking in the correct contexts.

[5]This seems to violate the Freezing Condition on movement which says that a moved constituent becomes an island for movement. Caha (2009, 47) discusses Freezing and argues that what holds is the *Criterial Freezing* Condition of Rizzi (2007) that forbids movement of a category X that has already reached a criterial position, but permits subextraction out of X. The peeling theory of movement is thus another example where the Criterial Freezing Condition holds.

(16)



The final assumption concerns spell-out of the case markers. I assume that the spell-out rules must be able to distinguish between checked and unchecked cases. As will be shown below, the distinction correctly predicts which spell-out rules have to apply. Furthermore, the distinction is crucial to allow pied-piping in case movement. In what follows, I will label checked cases with a superscript ✓, as exemplified in (16). With these assumptions in place, I turn to two examples of overt case stacking.

3.2 Overt case stacking: Huallaga Quechua and Ngiyambaa

3.2.1 Huallaga Quechua

The first instance of case stacking comes from Huallaga Quechua. The relevant example is given in (17).

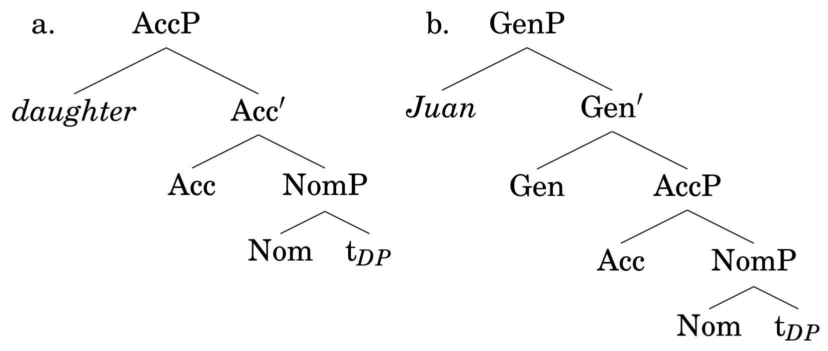
(17)

hipash-nin-ta	kuya-:	Hwan-pa-ta
daughter-3POSS-ACC	love-1	Juan-GEN-ACC

‘I love Juan’s daughter.’ (Plank 1995, 47)

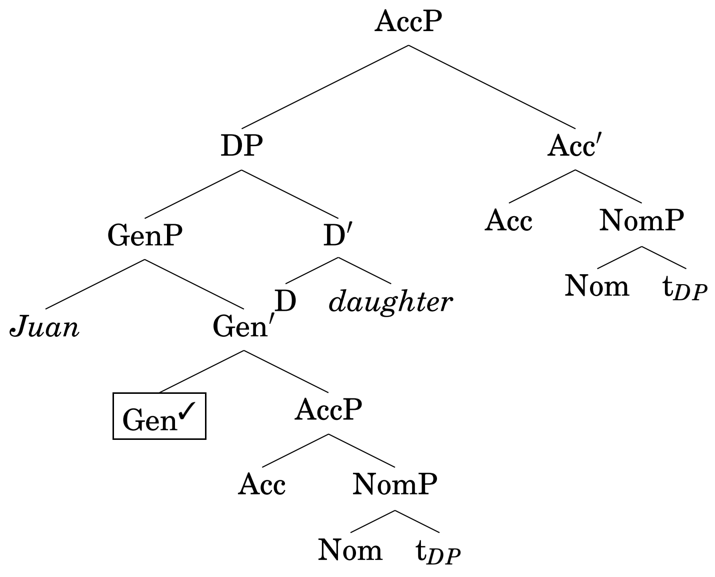
In (17), the possessum *daughter* is the direct object and bears the accusative. The possessor *Juan* bears the genitive and additionally the accusative in agreement with the possessum. (18) shows the KPs of the relevant nominal phrases *daughter* and *Juan*. The DPs have moved to the highest KP since, in Quechua, case markers are suffixes.

(18) a. b.



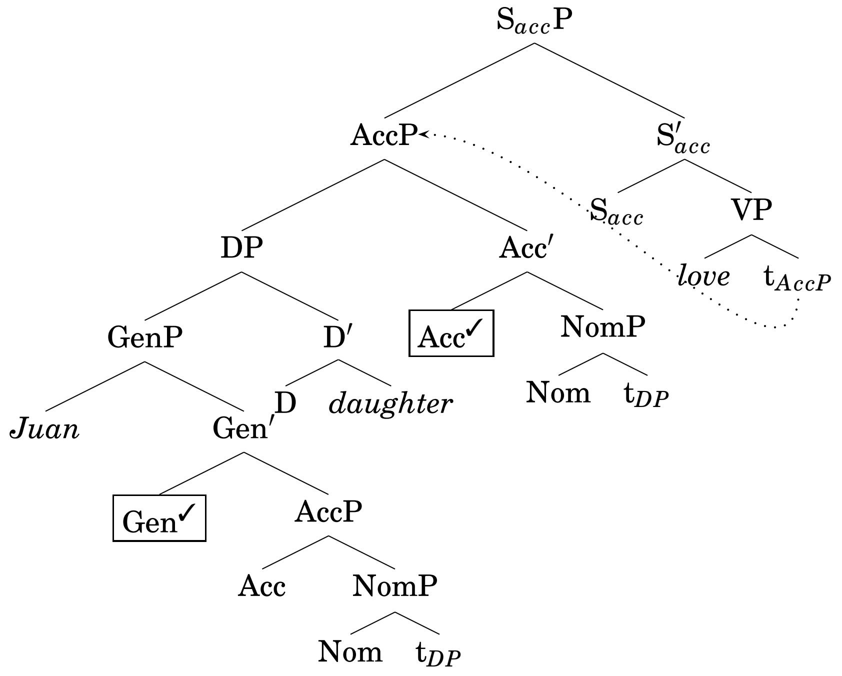
By assumption, the possessor in (18b) is merged as the specifier of the DP in (18a), where the genitive case is checked by D. The structure is shown in (19).

(19)



This entire AccP is merged in the object position of *love* and moves for case reasons into the specifier of a case checking head S_{acc} above VP.

(20)



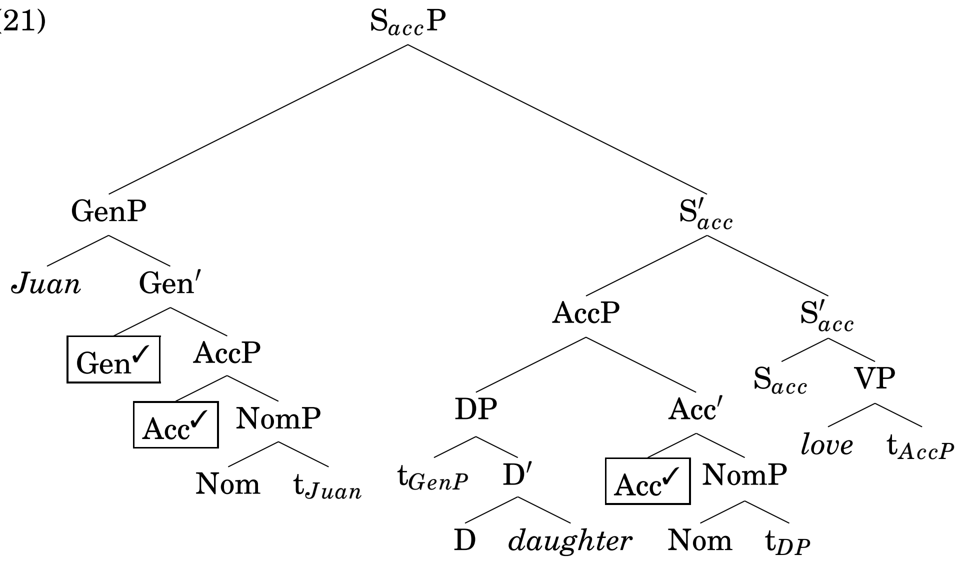
Now, since Quechua is a case stacking language, the AccP of the possessor *Juan* must move into Spec-S_{acc} as well. (Note that by assumption, Spec-S_{acc} allows multiple specifiers.) In the end, both the genitive and the accusative marker should show up on the possessor DP. In order to get this result, I suggest that movement of a KP in Quechua to a target case position involves pied-piping of higher KPs.[6] Pied-piping of the GenP in (20) is shown in (21). Note that the accusative can be checked by S_{acc} since the only case head between Acc and S_{acc} is Gen, which has already been checked and thus does not intervene for case checking of Acc.

[6]Note that a mechanism of pied-piping must be available independently for standard cases of pied-piping such as (i), where an embedded wh-phrase drags along the dominating DP.

(i) Whose mother was born in England?

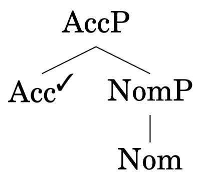
I will remain silent about the concrete implementation of pied-piping in a nanosyntactic framework and simply assume that some mechanism is available. See Ross (1967) for the phenomenon and Heck (2004) and references cited therein for theoretical approaches to pied-piping.

(21)

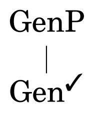


The resulting syntactic structure has to undergo spell-out now. The relevant spell-out rules for Huallaga Quechua are given in (22).

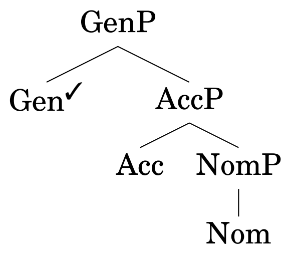
(22) a. /ta/ ⇔



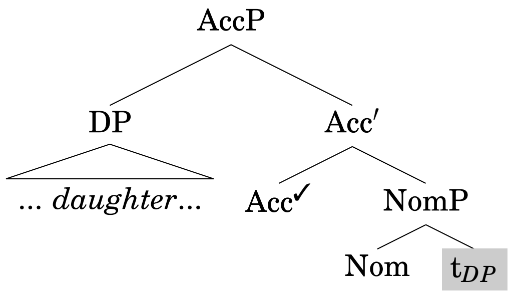
b. /pa/ ⇔

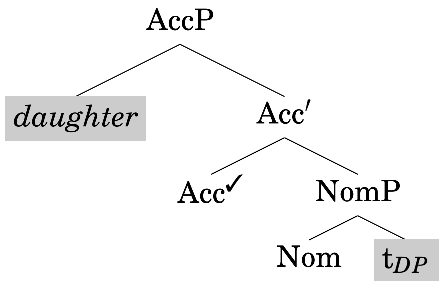


c. /pa/ ⇔

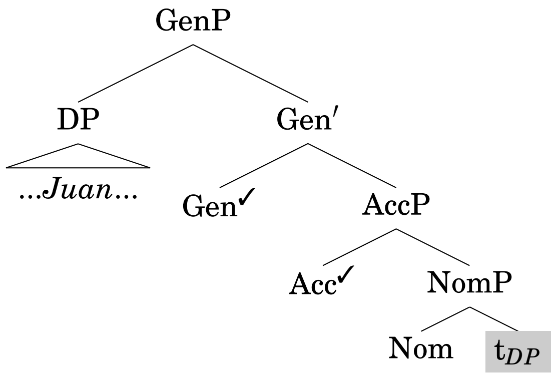


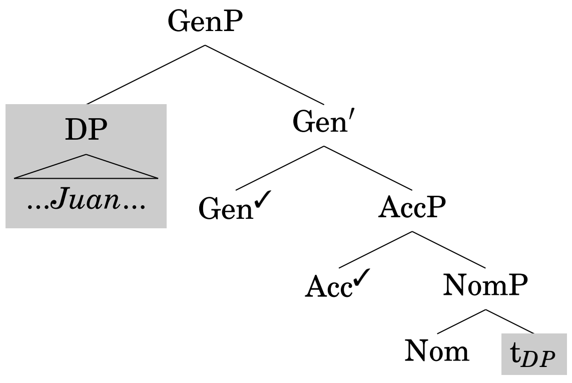
(22a) is the rule for the accusative marker */ta/*. Note that the spell-out rule can only apply to checked accusative. This guarantees that case stacking only applies in configurations where two cases have been checked. Otherwise we would expect Quechua to have compound case markers. The rules in (22b) and (22c) are two rules for the genitive marker */pa/*. The rule in (22c) is a general rule, which applies whenever there is no case stacking. The rule (22b) is the rule for the case stacking configuration in (21). When (22a) applies due to case checking of the embedded AccP, the spelled-out AccP becomes invisible and only the checked case feature Gen remains to be spelled out. (Note that (22c) cannot spell out this structure due to the Anchor Condition.) At this point, the rule (22b) can apply. The spell-out of the two DP's is shown in (23)– (24).

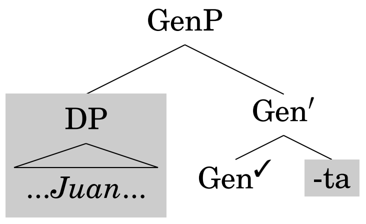
(23) a. 

b. ⇒ Spellout DP


c. ⇒ Spellout AccP (22a)
daughter-ta

(24) a. 

b. ⇒ Spellout DP


c. ⇒ Spellout AccP (22a)


d. ⇒ Spellout GenP (22b)
Juan-pa-ta

Finally, it should be noted that case stacking in Huallaga Quechua can only occur if the possessum and the possessor are separated. This follows directly from the analysis above: Movement of the possessor to Spec-S_{acc} will lead to case stacking on the one hand, but also to separation from the possessum on the other hand. Once the two are separated, the possessum can move freely as a remnant category to its target position.[7]

3.2.2 Ngiyambaa

While Quechua exhibited an instance of case stacking where a case high on the case hierarchy (accusative) stacks on a low case (genitive), Ngiyambaa exhibits an instance of case stacking where a low case stacks on a high one. The relevant example is given in (25).

(25)	ŋadhu	giyanhdha-nha	ŋidji-la:	winar-gu-dhi	miri-dji
	1SG-NOM	fear-PRES	this.CIRC-EST	woman-DAT-CIRC	dog-CIRC

‘I am frightened of this woman’s dog.’

(Donaldson 1980, Schweiger 2000, 258)

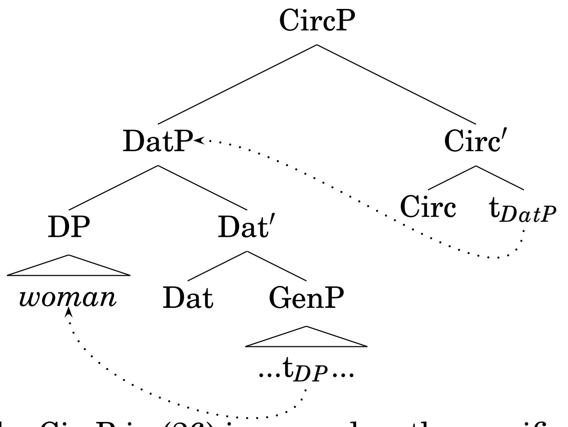
In (25), the possessor *woman* bears not only the possessive case dative but additionally the circumstantive of the possessum *dog*.[8] Since the circumstantive is a semantic case, it must be lower on the case hierarchy than the dative. For the nanosyntactic case theory, this means that the circumstantive is merged higher than the dative. To keep the derivations as simple as possible I will assume that the circumstantive is directly above the dative in the case sequence. Nothing hinges on this assumption.

The derivation of (25) begins with the possessor. The possessor is generated as a CircP since this case needs to be checked later. Within the CircP, the DP moves into the specifier position of the possessor case dative. Afterwards, the DatP moves into Spec-CircP. This movement is necessary because the dative must be checked by the D head in the possessive DP. If the DatP did not move, the unchecked circumstantive, which cannot be checked by D, would intervene for case checking of Dat. The structure is given in (26).

[7] In other case stacking languages, possessor movement might be banned due to other constraints. On the other hand, there are languages without overt case stacking that allow the separation of possessor and possessum, e.g. Udmurt (see Assmann et al. 2014, 472). At this point, I cannot offer a full-fledged solution to this problem. First, the correlation between case stacking on possessors and possessor movement needs to be studied in more detail. Then, it has to be examined whether the generalization to be found is compatible with the nanosyntactic approach to case stacking presented in this paper. I leave these issues to future research.

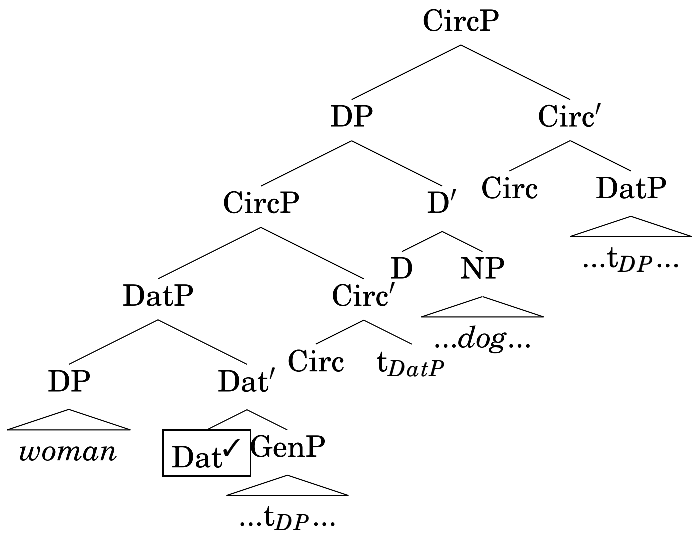
[8] The circumstantive marks the dog as being the reason for the fear of the woman.

(26)



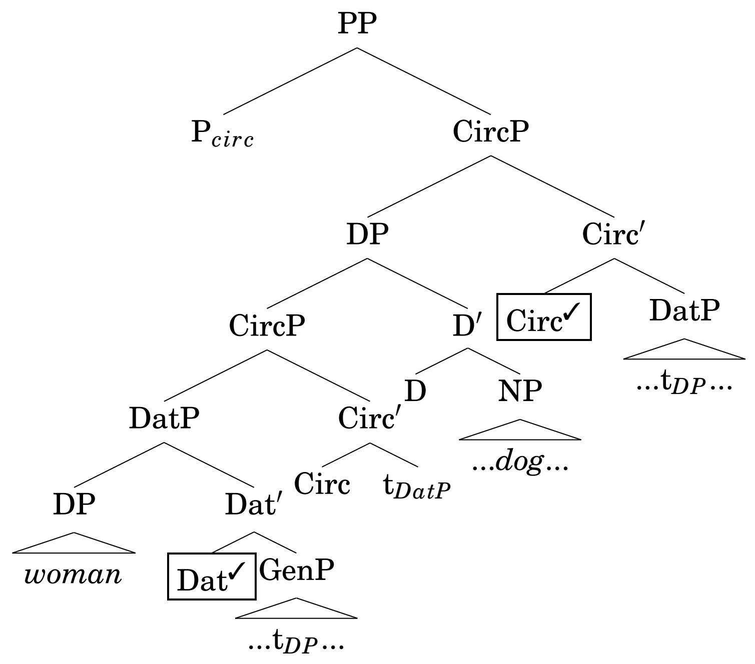
Next, the CircP in (26) is merged as the specifier of *dog*, where the possessor case dative is checked by D.

(27)



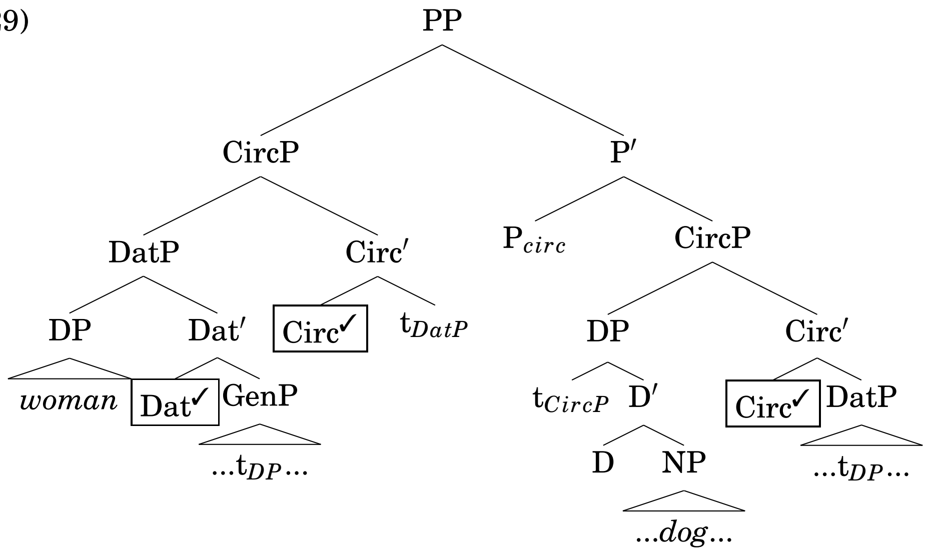
By assumption, semantic cases are checked by empty prepositions. Again, nothing hinges on that. The CircP in (27) is merged as the complement of such a preposition and gets its Case checked.

(28)



Afterwards the CircP of the possessor moves to Spec-PP and the Circ head gets checked.

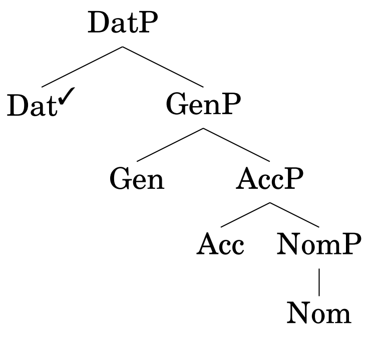
(29)



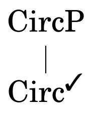
Finally, the structure must be spelled-out. Like in Quechua, all checked cases must be realized separately. The spell-out rules for the case markers in (29) are given in (30). Again we have at least two lexical entries for the circumstantive marker: (30b) applies in case stacking configurations, while (30c) applies in non-stacking configurations.[9]

[9]Note that the markers /dhi/ and /dji/ in (25) are allomorphs. The rules in (30b) and (30c) are meant to stand for the abstract morpheme.

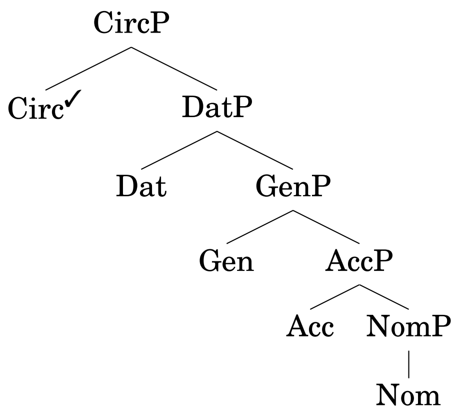
(30) a. /gu/ ⇔



b. /dhi/ ⇔

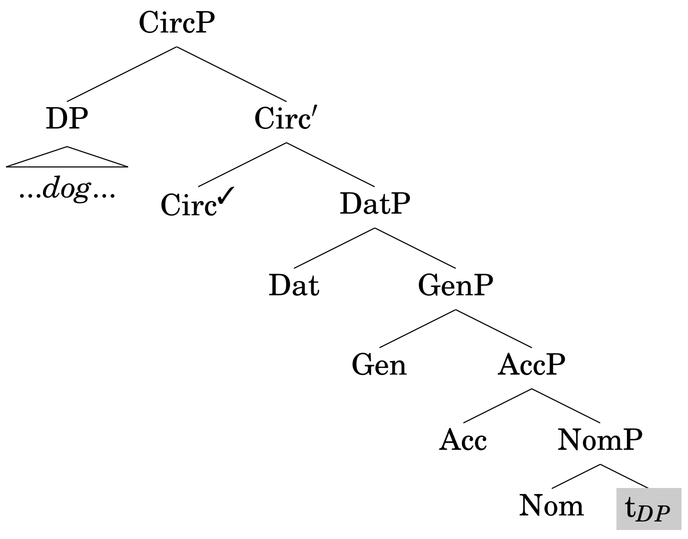


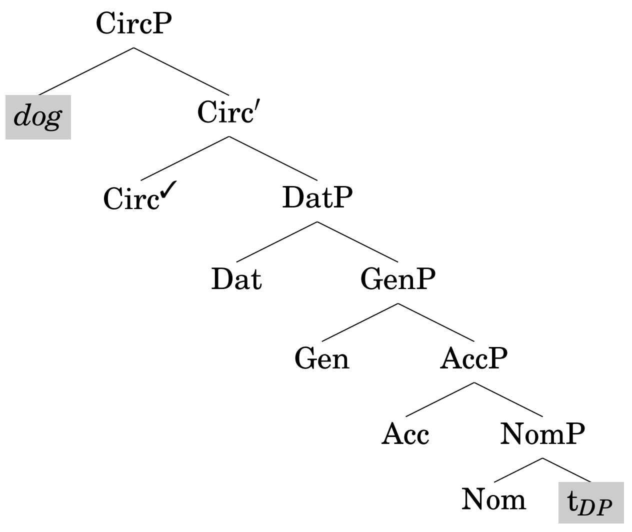
c. /dhi/ ⇔



The spell-out proceeds similar to the spell-out in Quechua. We start with the possessum phrase.

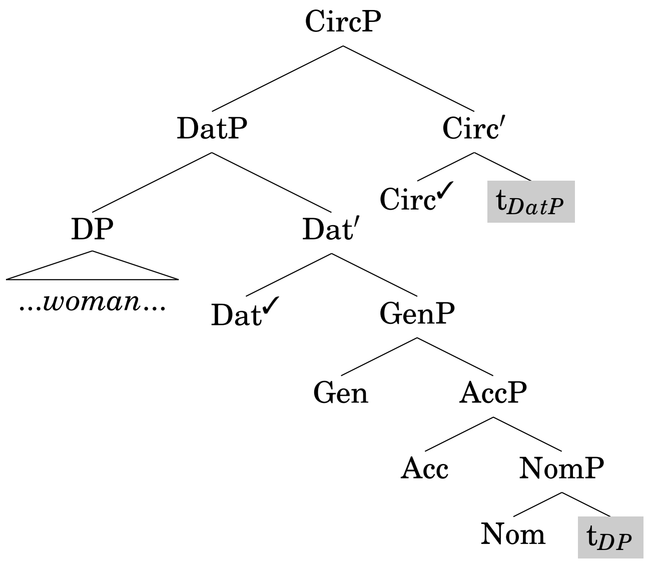
(31) a. ⇒ Spell-out DP



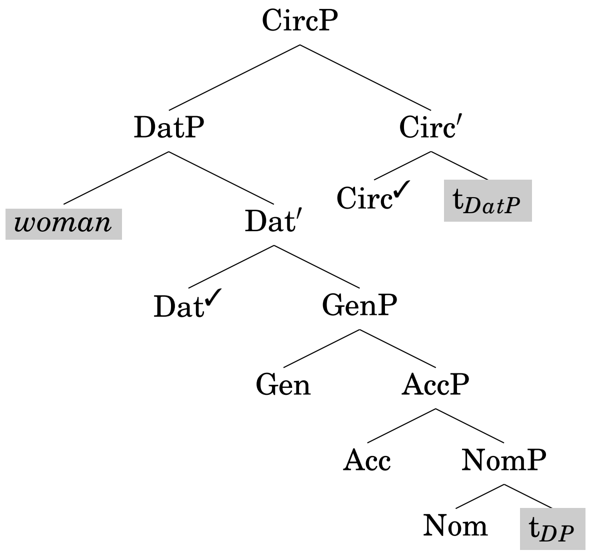
b.  ⇒ Spell-out CircP (30c)

c. *dog*-dhi

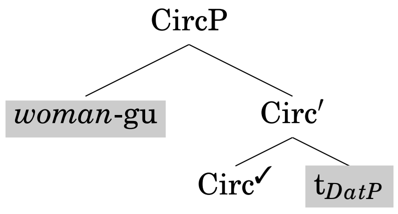
First, the DP *dog* is spelled out. Afterwards, the rest of the CircP is realized by the circumstantive marker /dhi/ according to (30c). The spell-out of the possessor is shown in (32).

(32) a. 

b. ⇒ Spell-out DP



c. ⇒ Spell-out DatP (30a)



d. ⇒ Spell-out CircP (30b)
woman-gu-dhi

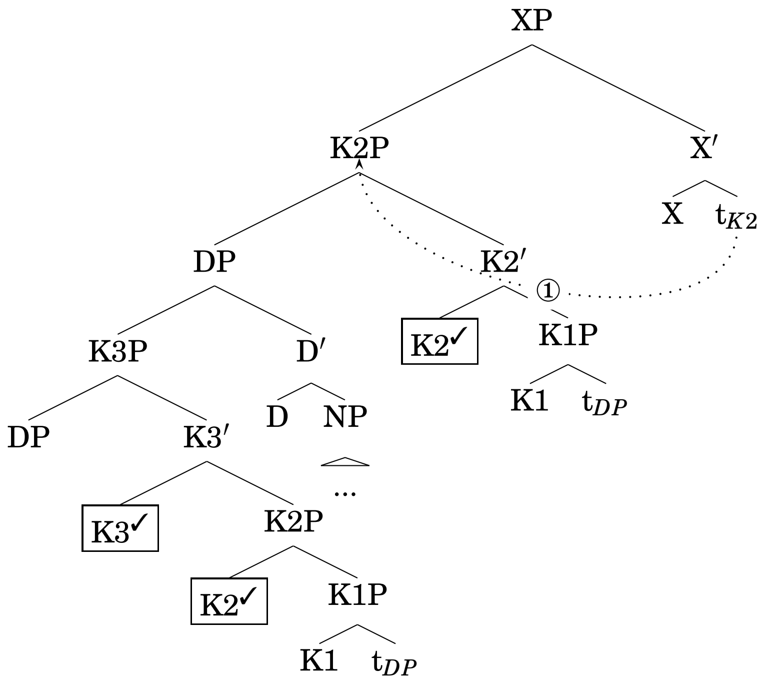
Again, the DP *woman* is spelled-out first. Then, the rule in (30a) can apply, because the dative case has been checked in syntax. The remaining CircP must be spelled out by rule (30b).

3.3 Discussion

This concludes the analysis of overt case stacking. The analysis essentially builds on the idea that case stacking comes about by multiple movement of KPs to the specifier of one case checking head. Overt case stacking results, if movement to a second case position pied-pipes the entire KP and if the two checked cases are realized with different markers.

There are, however, two alternatives to the present analysis. The first one involves case checking *in situ*. The assumption here is that a head checking case C can check any C head in its specifier. Therefore, multiple case movement is not needed. The abstract scheme is given in (33).

(33)



In (33), the head X checks the case K2. The higher K2P moves to Spec-X. In this position, both the higher and the lower K2P can be checked.

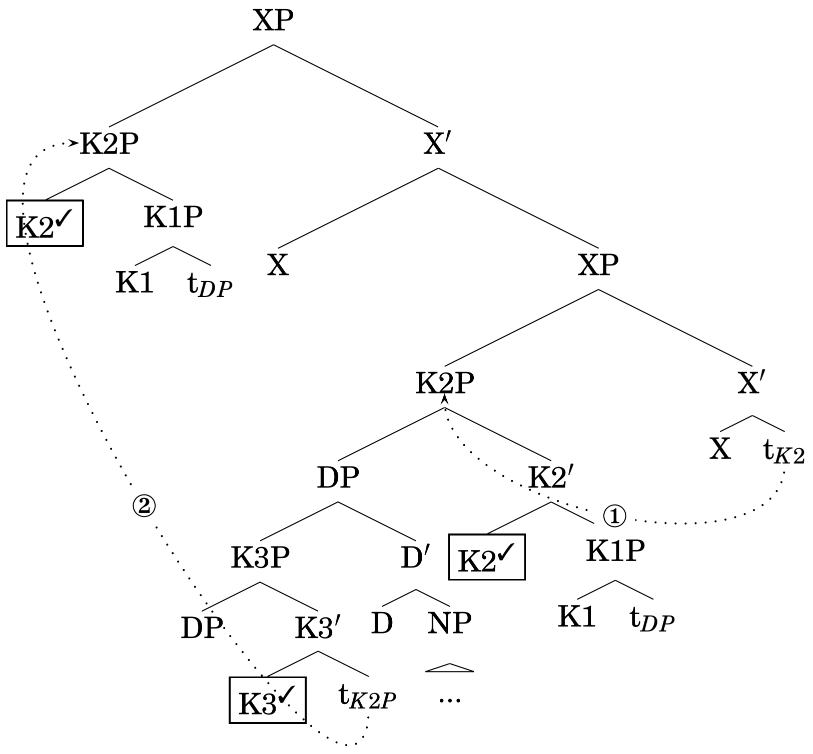
A potential problem with this approach concerns parametrization. In the analysis developed in section 3.1, the parameter between case stacking and non-case stacking languages involves the number of possible specifiers of a case checking head: In case stacking languages, a case checking head can have multiple specifiers. In languages that do not exhibit case stacking, only one specifier is allowed.[10] In the alternative analysis in (33), however, the parameter concerns a syntactic principle, namely in which configurations a head can check case: In case stacking languages the entire KP in specifier position, including embedded KPs, is visible to the case checking head, but in non-case stacking languages, only the highest KP is visible. While the parameter in the multiple specifier analysis can be modeled as a difference in the properties of a syntactic head, the *in situ* alternative requires the parametrization of a syntactic principle.

However, assuming that the Borer-Chomsky-Conjecture about linguistic variation (all variation is restricted to the lexicon, primarily to properties of inflection, see Chomsky 2001; Borer 1984; Baker 2008) also holds for the nanosyntactic approach, the latter way of parametrization is less than desirable.

A second alternative to the present multiple specifier approach to case stacking involves the presence of multiple case checking heads. The alternative is sketched in (34).

[10]This is reminiscent of the distinction between single and multiple wh-movement, as in English vs. Bulgarian (see Richards 1997).

(34)



First of all, it should be mentioned that the analysis in (34) does not represent the generalization about case stacking that the embedded DP receives the same case as the dominating DP. In (34), the embedded and the dominating K2P are checked by two different heads. It is more or less a coincidence that they are identical.

It is also more difficult in this approach to account for restrictions of case stacking. In many case stacking languages the number of cases that can stack is limited, e.g. in Thalanyii in (35) (Austin 1995, 373). In Thalanyii, at most two case markers can show up.

(35)

juma	jirrilarri-a	thuthu-wu	nganaju-wu	yakan-ku-wu
child.ABS	be.afraid-PRES	dog-DAT	I.DAT-DAT	spouse-DAT-DAT

‘The child is afraid of my wife’s dog.’

While the number of specifiers can be determined locally within a phrase, a restriction of the number of identical phrases requires a less local principle. Again, it seems that the multiple specifier approach is superior.

This concludes the discussion of two possible alternatives. In the next section, I will turn to an instance of abstract case stacking in Udmurt and show that it can be derived with the same means as overt case stacking.

4 Udmurt: an instance of abstract case stacking

As shown in Assmann et al. (2014), Udmurt exhibits a case split on the possessor between genitive and ablative case. The concrete case of the possessor depends on the case of the possessum: If the possessum bears accusative case, the possessor is marked for ablative case. In all other cases, the possessor bears genitive case. This is an interesting problem for theories of case assignment because, derivationally, the case of the possessum is determined after the case of the possessor. This leads to a look-ahead problem. Assmann et al. (2014) argue that this case split can be reanalyzed as an instance of abstract case stacking: The possessor always receives the same possessive case (genitive) but additionally receives the case feature of the dominating DP. Certain morphological constraints and rules lead to a postsyntactic manipulation of the case features which change the contexts for vocabulary insertion. Concretely, if a possessor bears genitive and accusative case, its case features must be realized by the ablative marker. All other case combinations are realized with the genitive marker. In what follows, I show that the case split can be analyzed in the nanosyntactic approach as case stacking as well. This corroborates the case stacking analysis of Udmurt since the main idea of the analysis seems to be independent of the concrete morphosyntactic framework. The relevant data are given in (36).

(36) a. so-len/*leš anaj-ez siče ug diśaśki
he-GEN/ABL mother-3SG such dress NEG.PRES.3
'His mother does not dress in such a way.'
(Edygarova 2009, 105)

b. so-*len/leš eš-s-e ažži-śko
he-GEN/ABL friend-3SG-ACC see-PRES.1SG
'I see his friend.' (Edygarova 2009, 101)

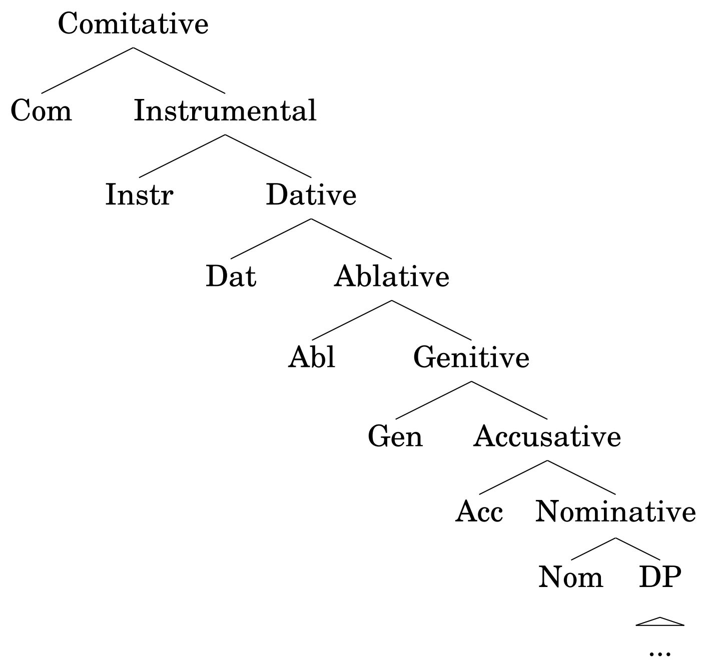
c. mon [Petyr-len/*leš puny-jez-leš] mözm-is'ko
1SG Peter-GEN/ABL dog-3SG-ABL miss-PRES.1SG
'I miss Peter's dog.' (Assmann et al. 2014, 453)

d. Petyr Masha-len apaj-ez-leš puny-z-e zhug-i-z
Peter Masha-GEN sister-3SG-ABL dog-3SG-ACC beat-PST-3SG
'Peter has beaten Masha's sister's dog.'
(Assmann et al. 2014, 454)

(36a) shows that the possessor bears genitive case if the possessum is the subject. In (36b) the possessor bears ablative because the possessum is marked for accusative case. (36c-d) show that the ablative on the possessor in (36b) is indeed due to the accusative of the possessum: if the object is marked with any other case, even the ablative as in (36c), the possessor receives genitive case. (36d) shows that in multiple possessor constructions, only the highest possessor is marked ablative in line with the generalization that the DP directly dominating an ablative possessor must bear accusative.

The first assumption for an account of the case split in Udmurt concerns the ablative case feature. In Caha (2009, 16), the ablative and the locative case are not included in the universal case hierarchy. Thus, it is reasonable to assume that their position in the case hierarchy is language-specific. I assume that the ablative in Udmurt is located right above the genitive in the functional sequence (cf. Caha 2009, 213 for a similiar case hierarchy for Armenian). This captures the fact that the ablative in Udmurt functions as the default semantic case that occurs whenever there is no more specific semantic case feature. (Nominative, accusative, and genitive are structural cases in Udmurt.) This will be important in section 4.2 when the case split is discussed.

(37)



Next, just as in Huallaga Quechua and Ngiyambaa, objects are base-generated with an appropriate case. For Udmurt that means that ablative assigning verbs take Abl-PPs as complements (assuming that semantic cases are checked by prepositions), accusative assigning verbs take AccPs, etc. Furthermore, all case markers in Udmurt are post-nominal (including adpositions), thus, following the general logic of the nanosyntactic approach, the DP in Udmurt moves into the highest case layer in the respective KP.

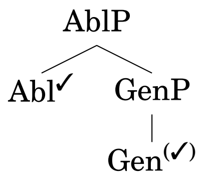
To simplify the derivations below, I assume that all cases except the accusative and the nominative are checked in situ. (Nothing hinges on that.) The accusative and nominative case are checked by movement to the specifier position of case checking heads S_{acc} and S_{nom}.

In order to derive the effects of case stacking and to stay as close as possible to the minimalist analysis in Assmann et al. (2014), I assume that both case checking heads S_{acc} and S_{nom} are present in every active transitive structure. (In the passive and in intransitive clauses, S_{acc} is missing.) Like in other case stacking languages, these two case heads are able to check more than one case.

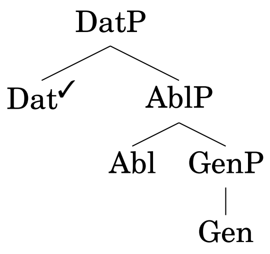
Finally, the reason why we don't observe overt case stacking in Udmurt is that there is no pied-piping in Udmurt. Instead movement in order to check a second case strands higher KP-shells including the DP. Since the case markers in Udmurt are suffixes and as such morphologically dependent on a DP, a moved KP without a DP cannot be realized. Instead, only the stranded KP, which contains the DP, is morphologically realized. This leads to the absence of overt case stacking in Udmurt. The lexical entries for Udmurt are given in (38). Despite the ablative, only one other semantic case is listed in (38), which should illustrate the general scheme. Crucially, the ablative is the most specific semantic case and can only be inserted in the environments $[_{AblP}$ D $[_{GenPC}]]$ and $[_{GenPC}]$. Thus, the ablative will always out-rank the other semantic cases due to the Elsewhere Condition (7). This is in line with the default character of the ablative.

(38) *Lexical Entries*

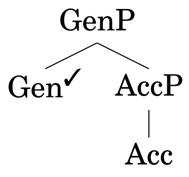
a. /leš/ ⇔



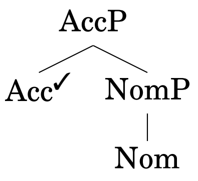
b. /li/ ⇔



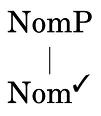
c. /len/ ⇔



d. /ez/ ⇔



e. /∅/ ⇔

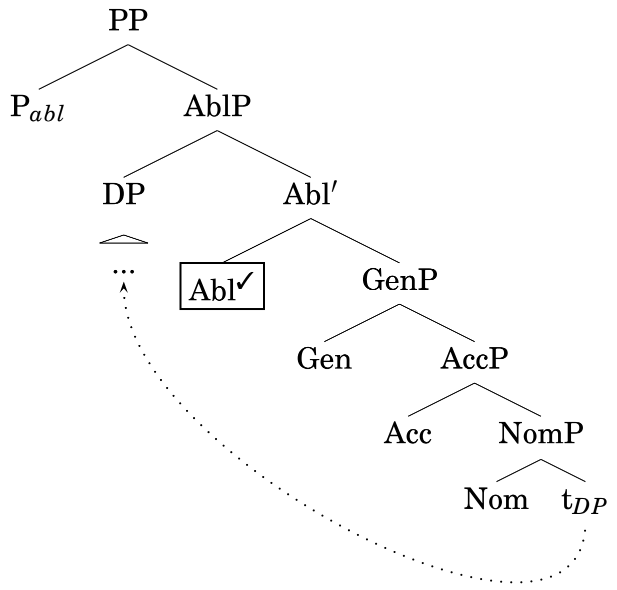


Note that the marker /leš/ in (38a) is insensitive to whether the genitive case is checked or not. Thus, it can be said, that the entry is underspecified for whether Gen is checked or not. This will be important for the derivation of the case split on the possessor in section 4.2. But first, 4.1 shows how case checking of syntactic arguments in Udmurt proceeds in general.

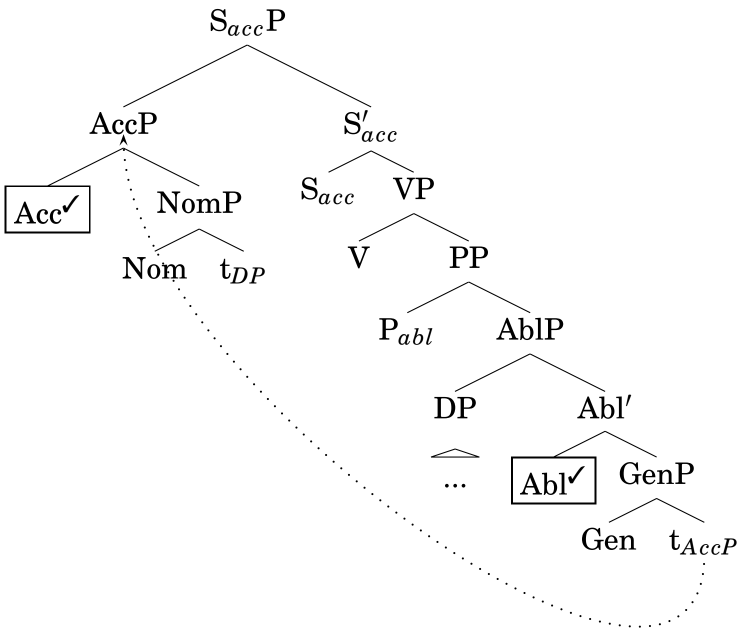
4.1 Case checking of syntactic arguments

The first derivation I would like to illustrate is a structure where a verb takes an ablative object. First, a DP with an AblP on top is base generated. Since the ablative is a suffix, the DP moves above AblP. This AblP is merged with an empty preposition P_{abl}, which checks the case head Abl.[11]

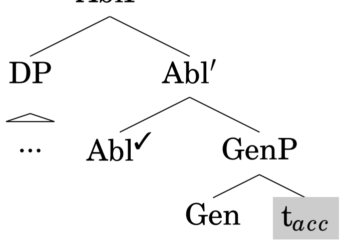
[11]Note that the ablative is a semantic case. By assumption, all semantic cases are checked by prepositions. Nothing hinges on that.

(39)


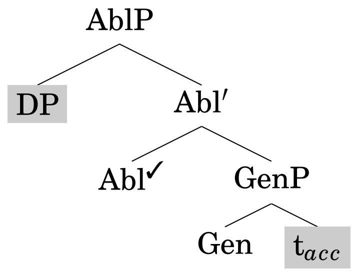
Next, the verb and the obligatorily present case checking head S_{acc} are merged. The AccP inside the PP moves to the specifier of S_{acc}, stranding AblP including the DP. The feature of AccP is checked.

(40)


The stranded AblP is now morphologically realized using the rule in (38a).

(41) a.


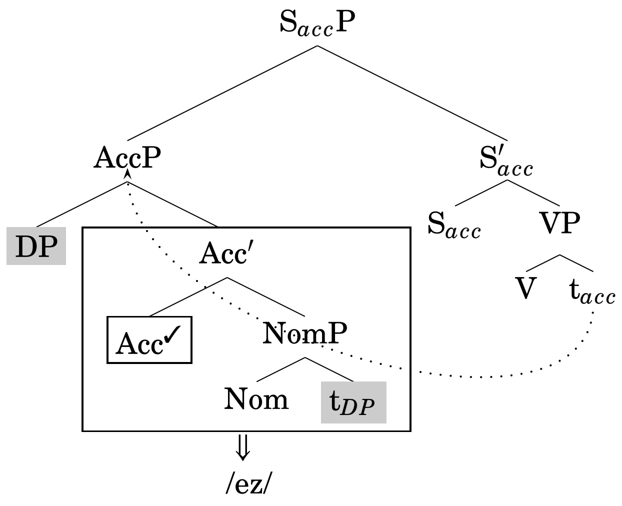
b. ⇒ Spellout DP



c. ⇒ Spellout AblP (38a)
DP-leš

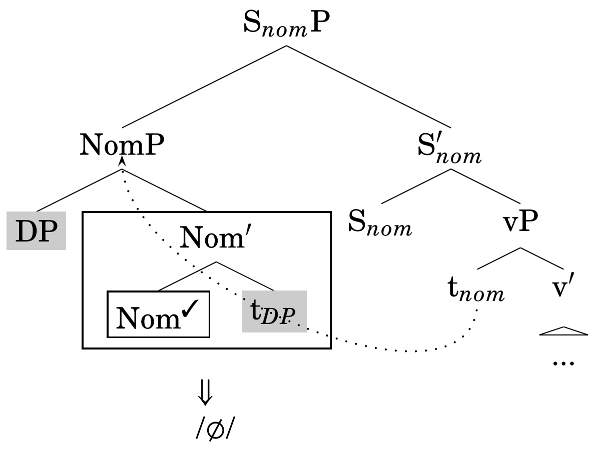
Next, we turn to a derivation with an accusative object. Here, the DP is base-generated with an AccP on top. Again, the verb and the case checking head S_{acc} are merged and AccP moves to the specifier of S_{acc}, stranding nothing behind. The case head Acc of AccP is checked and AccP is morphologically realized by /ez/ according to the rule in (38d).

(42)



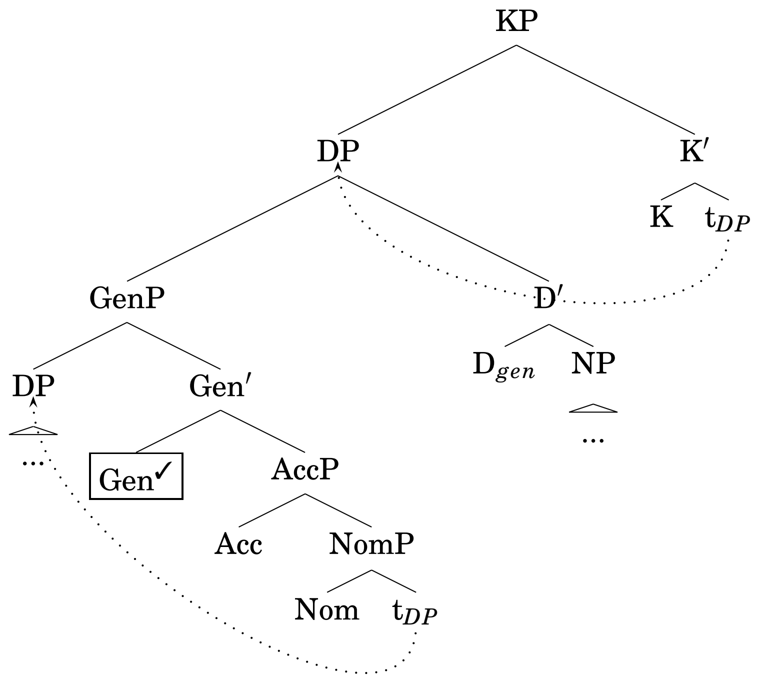
Similarly, subjects are base-generated with a NomP on top and check their case by movement to the specifier of a case checking head S_{nom}. NomP is realized by /∅/ according to (38e).

(43)



Finally, genitive case checking proceeds as follows. A possessor with a GenP on top is base-generated as the specifier of its possessum. The D-head of the possessum checks the genitive case of the possessor. The entire possessive DP is moved to the specifier of the highest case layer (here KP) of the possessum. In the following section, KP will be replaced by concrete case layers and we will see how this affects further case assignment and case feature realization.

(44)

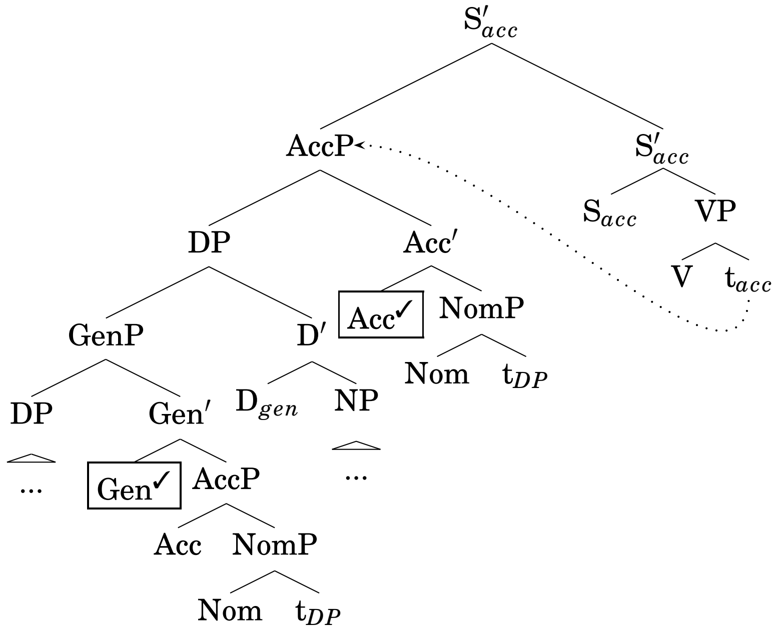


So far, we have seen simple instances of case checking. No case stacking applied so far. The next section shows how abstract case stacking, that is, multiple case movement, can derive the case split in Udmurt. Since Udmurt does not exhibit overt case stacking, one KP of an argument – the one containing the DP – is spelled-out.

4.2 The case split on the possessor

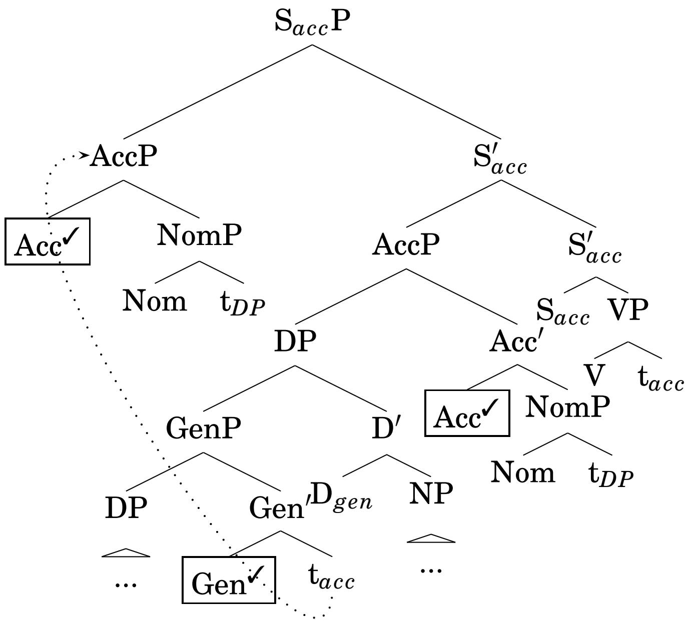
Let's begin with the case where a DP is the possessor of an accusative argument. As in the derivation in (42), the entire AccP, this time including the possessor in its specifier, is moved and checks its accusative case feature against the head S_{acc}.

(45)



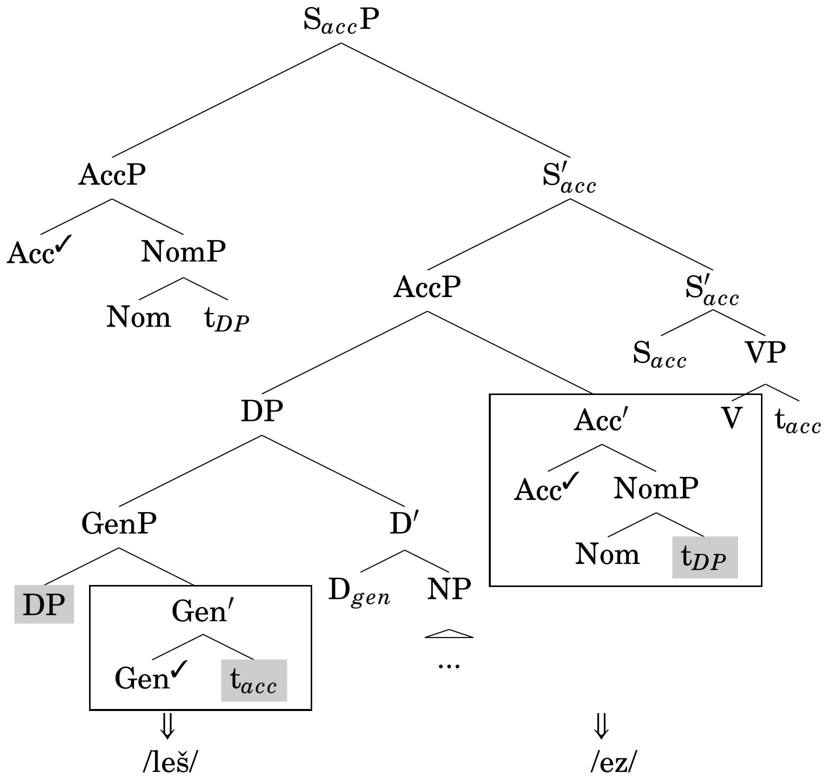
Now, the AccP of the possessor can move to Spec-S_{acc} and checks its accusative case feature.

(46)



In Spell-out, both the case features of the possessum and the possessor must be realized. The case of the possessum is straightforward: it must be /ez/, just as in (42). The case marker of the possessor must by assumption realize the stranded GenP. (The moved AccP does not contain a DP.) Since the genitive marker in (38c) does not fit (due to the Anchor Condition in (8)), the ablative case rule in (38a) must do the job.

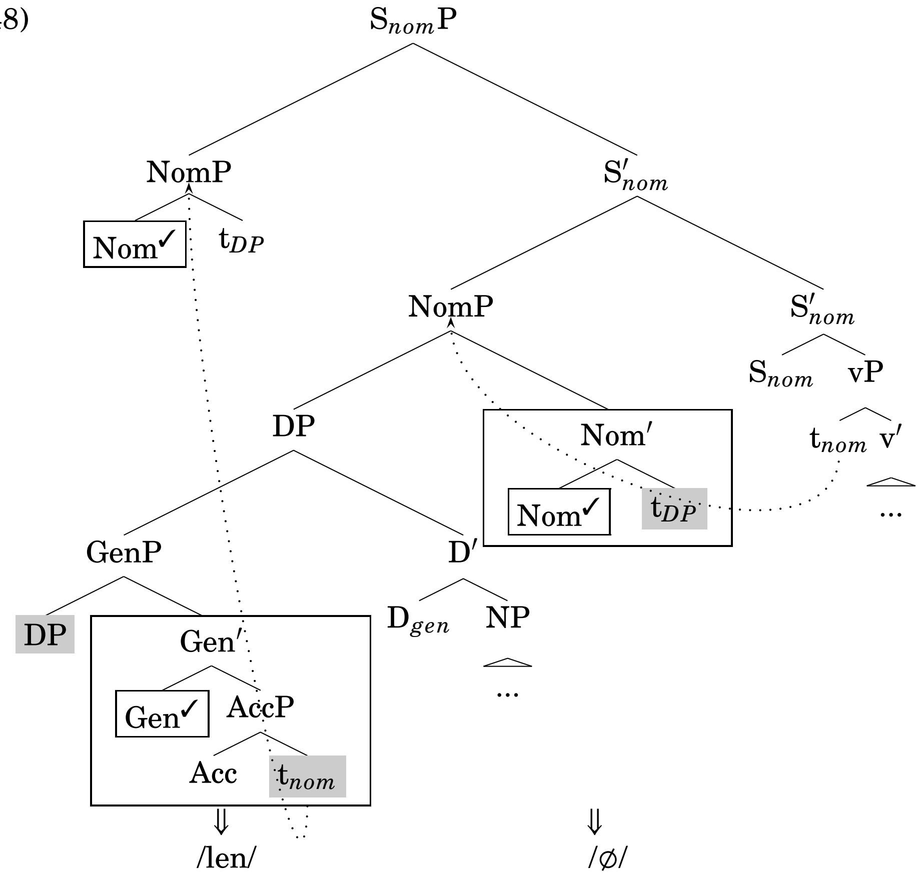
(47)



This derives one part of the case split in Udmurt. Due to multiple case checking of the possessor, the configuration inside the GenP of the possessor changes in a way that only the ablative marker can be inserted. So abstractly, Udmurt exhibits case stacking. However overtly, Udmurt only realizes the stranded KP. Thus, there is no multiple case marking on the possessor, which points to overt case stacking.

If a DP is the possessor of a nominative possessum, the realization of the case features must involve other rules. In the syntax, the entire NomP including the possessor is moved and checks its nominative case feature against the head S_{nom} (cf. (43)). Next, the NomP of the possessor can move and check its nominative case feature. The case of the possessum must be realized by /∅/, analogous to (43). The possessor can, in contrast to (47), be realized by the matching genitive marker /len/ in (38c). The NomP that moved out of the possessor is not realized because it does not contain a DP.

(48)



This derives another part of the case split in Udmurt. Similar to the abstract case stacking of genitive and accusative in (47), we have abstract case stacking of genitive and nominative in the derivation in (48). But this time, the spell-out rule for the genitive marker applies.

The final part to derive the case split are configurations where case stacking is blocked. Not all possessors of objects are marked with ablative case. If the possessum bears a case other than accusative, the possessor is marked for genitive case. In what follows, I will show that this genitive marking is due to blocking of case stacking. As shown below, the configuration for case stacking is only given if the case of the possessum is accusative or nominative (cf. the derivations in (47) and (48)). This is partly due to the case sequence in (37): Possessors are always GenPs. In the case sequence in (37), the only KPs that can move out of this GenP are AccP and NomP. The KPs of the semantic cases are higher. Therefore, a semantic case cannot cannot stack on the genitive.

But assuming that the case checking heads S_{acc} and S_{nom} are always present in active transitive clauses, we could in principle have case stacking of genitive and nominative and genitive and accusative in structures where the possessum is a KP higher than accusative. This would lead to the same case split as derived in (47) and (48). To block case stacking here, I assume that, if a case checking head X has specifiers, it can only subextract a second KP from its highest specifier, not from its complement or inner specifiers.[12] The condition is formulated in (49).[13]

(49) In a configuration [$_{\text{XP}}$ α [$_{\text{X}'}$ ... X β]] X may subextract a category only from α.

If the possessum is a KP that is higher than accusative, the first case checking will target a position that is not in Spec-S$_{acc}$ or Spec-S$_{nom}$, respectively. Subsequent movement to any of these two position will strand the possessum DP, which contains the possessor. Thus, the possessor is not in the outermost specifier of S$_{acc}$ or S$_{nom}$. Therefore, the AccP or NomP of the possessor is not visible to S$_{acc}$ or S$_{nom}$, respectively, and case stacking is blocked.

The first configuration where case stacking is blocked is a configuration where the possessum bears a semantic case like, for example, the ablative. The structure is shown in (50).

[12]The idea resembles to some extent the concept of *cyclic expansion of the search domain* (see Béjar and Řezáč 2009, 49).

[13]This constraint might follow if it is assumed that the search domain of an attracting head X is subject to the strict cycle condition Chomsky (1973). Under a certain understanding of the SCC, a second search in the complement of X would violate the SCC, since it is not the root node that is affected. The configurations are shown in (i).

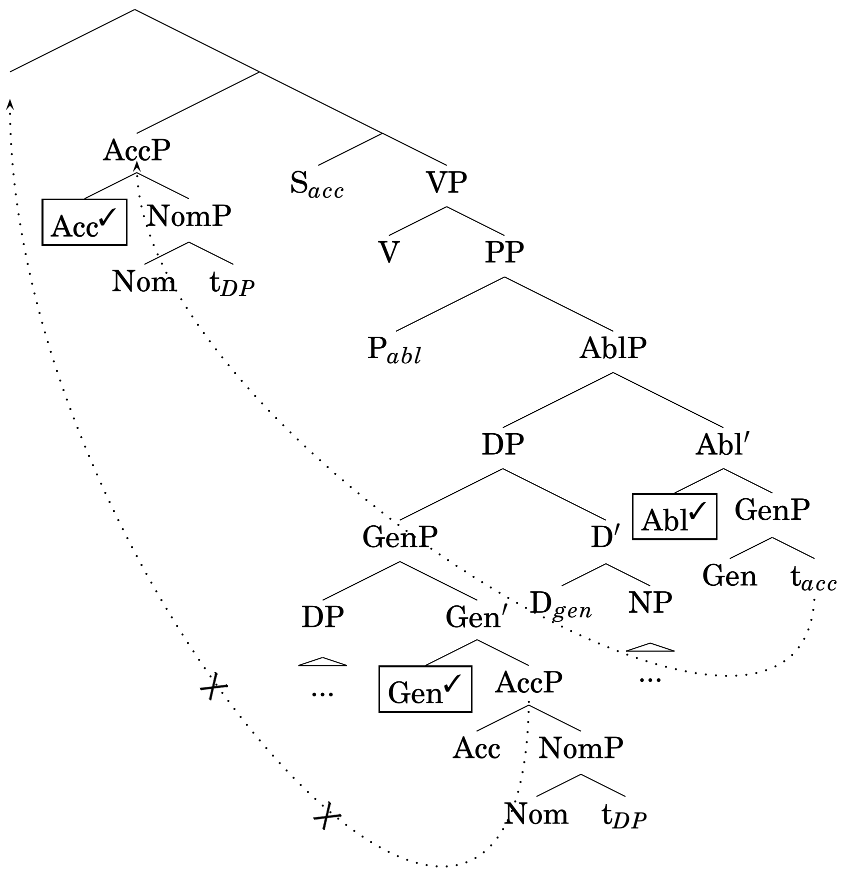
(i) a. $$\begin{tikzcd}[\text{XP} \; \text{X} \arrow[r, bend right] & {[}\; ... \; \alpha \; ... \;]]\end{tikzcd}$$

b. $$\begin{tikzcd}[\text{XP} \; \alpha \; [\text{X}' \; \text{X} \arrow[r, bend right, "\times\times"'] & {[}\; ... \; t_\alpha \; ... \;]]]\end{tikzcd}$$

c. $$\begin{tikzcd}[\text{XP} \; \alpha & {[}\text{X}' \; \text{X} \arrow[l, bend left] \; [\; ... \; t_\alpha \; ... \;]]]\end{tikzcd}$$

In (ia), there is no specifier in the XP. Thus the search of X for a matching goal in the complement affects the entire tree XP. In (ib), a specifier of X has been merged. Now, a search in the complement would violate the SCC since it would only affect a subtree. Therefore, as shown in (ic), X has to search in its outermost specifier.

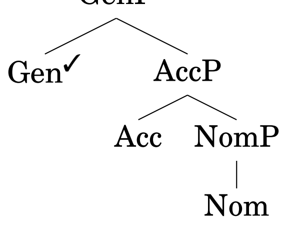
(50)



Here, the AccP inside the possessor cannot move to Spec-S_{acc} because the dominating KP of the possessum is not a specifier of S_{acc}. At the point when S_{acc} has attracted the AccP of the possessum to its specifier, only this AccP is visible to S_{acc}. But since this AccP does not contain another AccP, S_{acc} cannot attract a second specifier.

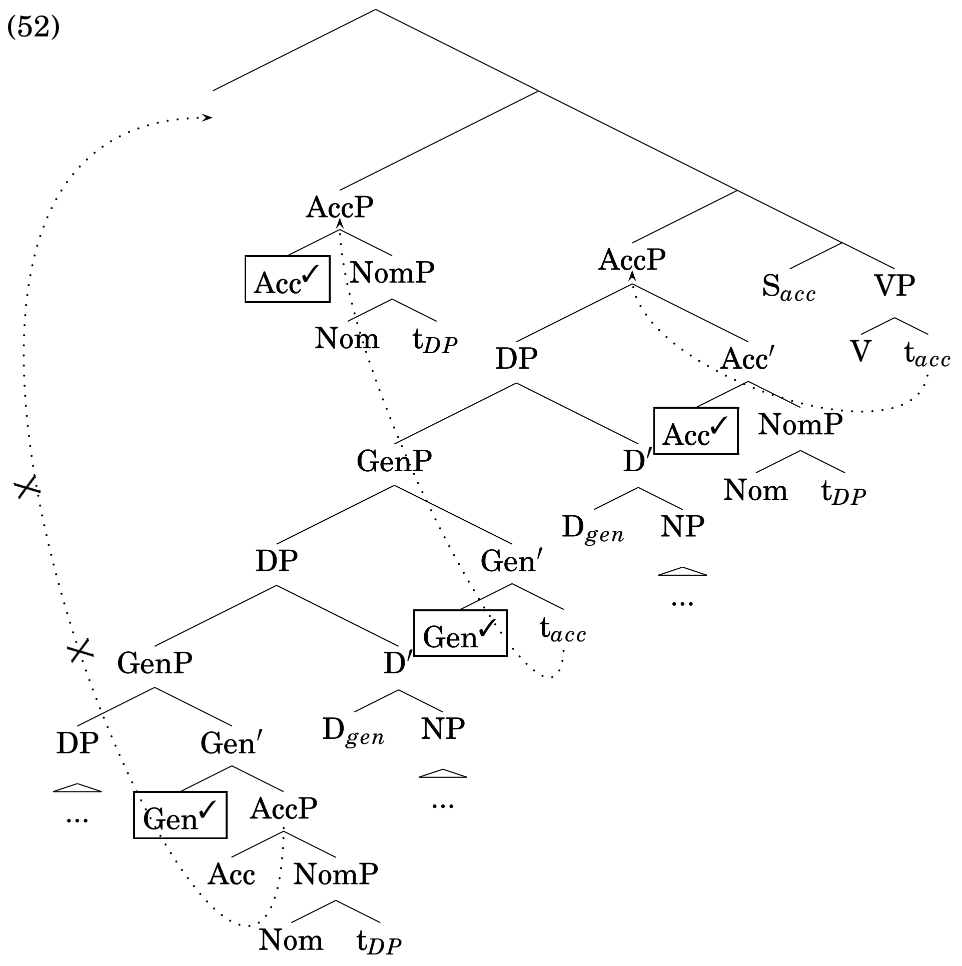
When it comes to spell-out of the possessor's case marker, none of the rules in (38) applies because the context is not given due to the Anchor Condition in (8). Therefore, we must add a third lexical entry for the genitive marker /len/, that applies when no case stacking takes place. The GenP of the possessor in (50) is realized with the genitive marker /len/, if we add the following lexical entry.

(51) /len/ ⇔



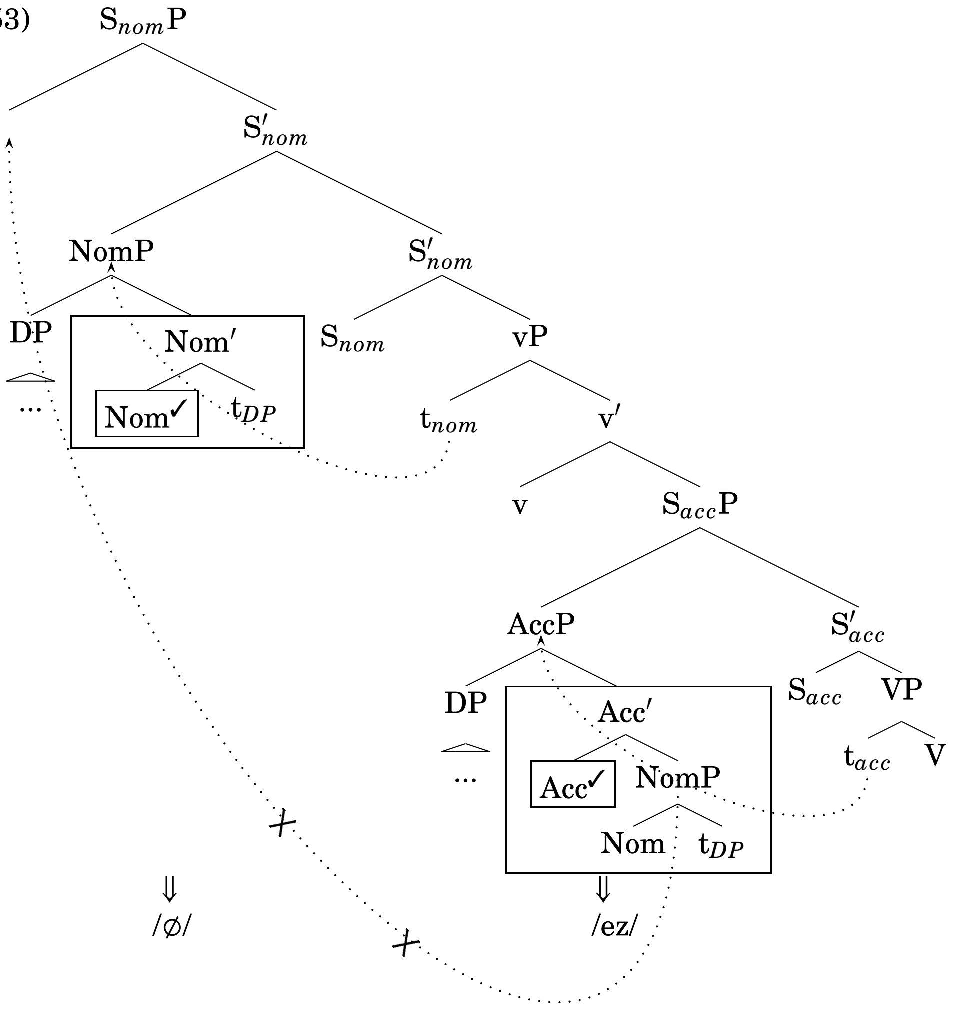
The next blocking configuration involves multiple possessors as illustrated in (52).

(52)



Similar to the derivation in (51), the AccP of the most embedded possessor cannot move to Spec-S_{acc} because its possessum, which is the higher possessor, is not in the highest specifier of S_{acc}. When the AccP of the higher possessor moves to Spec-S_{acc}, S_{acc} can only search in this AccP. But since it doesn't find another AccP, no case stacking takes place. The GenP of the lower possessor must be realized by the rule in (51), similar to the derivation in (50).

Finally, there is a third configuration where case stacking is blocked, namely in simple transitive sentences with a nominative subject and an accusative object. The reason is the following: S_{nom} first attracts the NomP of the subject, due to minimality. But then, the search domain of S_{nom} is restricted to this NomP. The NomP of the object is not visible to S_{nom}. The structure is shown in (53).

(53)


This concludes the case study of abstract case stacking. I have made a suggestion as to how the case split in Udmurt can be analyzed as abstract case stacking in a nanosyntactic approach. Case stacking in nanosyntax simply means that multiple heads inside one case sequence are checked by movement. This is identical to the derivations of overt case stacking in section 3. The difference between overt and abstract case stacking is whether multiple case movement involves pied-piping or not. In Huallaga Quechua and Ngiyambaa, movement of KPs involved pied-piping of higher and lower KPs, including the DP. Since the DP was pied-piped, all case suffixes could be realized. In Udmurt, however, further movement of KPs does not pied-pipe other KPs, including the DP. Assuming that the spell-out rules for case suffixes can only apply in the context of a DP, the lack of pied-piping causes leads to the structures, where only the stranded KP that contains the DP can be realized. Overtly, no case stacking shows. Finally, in some configurations, case stacking must be blocked. I have proposed that this is due to a constraint that only the outermost specifier of a head is visible to this head. This restricts the number of contexts for case stacking and correctly block case stacking in Udmurt when the possessum bears a case other than accusative or nominative.

5 Conclusion

In this paper, I made a proposal as to how case stacking can be modeled in a nanosyntactic approach to case assignment. The main idea is that a case checking head can attract more than one KP. Combining this with a mechanism of pied-piping such that higher or lower KPs tag along after the target KP, derives overt case stacking as shown for Huallaga Quechua and Ngiyambaa. If multiple case movement applies without pied-piping, abstract case stacking as in Udmurt can be derived. Blocking of case stacking is derived under the assumption that outer specifier positions can only be filled by categories from inside the outermost specifier.

The present analysis confirms the account in Assmann et al. (2014) that claims that Udmurt exhibits an instance of abstract case stacking. It was shown that this idea can be maintained even if a different morphosyntactic framework is used.

References

Assmann, Anke, Svetlana Edygarova, Doreen Georgi, Timo Klein and Philipp Weisser (2014): 'Case stacking below the surface: On the possessor case alternation in Udmurt', *The Linguistic Review* **31**(3-4), 447–485.

Austin, Peter (1995): Double Case Marking in Kanyara and Mantharta Languages, Western Australia. *In:* F. Plank, ed., *Double Case: Agreement by Suffixaufnahme*. Oxford University Press, New York, pp. 363–379.

Baerman, Matthew, Dunstan Brown and Greville G. Corbett (2005): *The Syntax-Morphology Interface. A Study of Syncretism*. Cambridge University Press, Cambridge.

Baker, Mark C. (2008): *The Syntax of Agreement and Concord*. Cambridge University Press.

Béjar, Susana and Milan Řezáč (2009): 'Cyclic Agree', *Linguistic Inquiry* **40**(1), 35–73.

Belletti, Adriana, ed. (2004): *Structures and Beyond. The Cartography of Syntactic Structures*. Oxford University Press, New York.

Bjorkman, Brownwyn (2013): The Syntax of Syncretism. *In:* S. Kan, C. Moore-Cantwell and R. Staubs, eds, *Proceedings of NELS 40*. GLSA, Amherst, Massachusetts, pp. 71–84.

Borer, Hagit (1984): *Parametric syntax: case studies in Semitic and Romance languages*. Foris, Dordrecht.

Caha, Pavel (2009): The Nanosyntax of Case. PhD thesis, University of Tromsø, Tromsø.

Chomsky, Noam (1973): Conditions on Transformations. *In:* S. Anderson and P. Kiparsky, eds, *A Festschrift for Morris Halle*. Academic Press, New York, pp. 232–286.

Chomsky, Noam (2001): Derivation by Phase. *In:* M. Kenstowicz, ed., *Ken Hale: A Life in Language*. MIT Press, Cambridge, Massachusetts, pp. 1–52.

Cinque, Gugliemo, ed. (2002): *Functional structure in DP and IP. The Cartography of Syntactic Structures*. Oxford University Press, New York.

Donaldson, Tamsin (1980): *Ngiyambaa*. Cambridge University Press, Cambridge.

Edygarova, Svetlana (2009): ‘Attributive Possession in Udmurt Language’, *Linguistica Uralica* **45**, 101–118.

Erlewine, Michael Yoshitaka (2013): ‘Dissociating the syntax and morphological realization of Kaqchikel Agent Focus’, *MIT Working Papers on Endangered and Less Familiar Languages* **8**, 25–50.

Heck, Fabian (2004): ‘On Certain Properties of Pied-Piping’, *Linguistic Inquiry* **40**(1), 75–111.

Hiraiwa, Ken (2001): Multiple Agree and the Defective Intervention Constraint in Japanese. *In:* O. Matushansky, A. Costa, J. Martin-Gonzalez, L. Nathan and A. Szczegielniak, eds, *Proceedings of the HUMIT 2000*. Vol. 40 of *UCLA Working Papers in Linguistics*, University of California, Los Angeles, pp. 67–80.

Matushansky, Ora (2008): Predication: a Case Study. *In:* F. Marušic and R. Žaucer, eds, *Studies in Formal Slavic Linguistics. Contributions from Formal Description of Slavic Languages 6.5*. Peter Lang, Frankfurt am Main, pp. 213–239.

McCreight, Katherine (1988): Multiple case assignments. PhD thesis, MIT, Cambridge, MA.

Merchant, Jason (2006): ‘Polyvalent case, geometric hierarchies, and split ergativity’, *Chicago Linguistics Society (CLS)* **42**, 47–67.

Nordlinger, Rachel (1998): *Constructive case: Evidence from Australian languages*. CSLI, Stanford.

Pesetsky, David (2013): *Russian case morphology and the syntactic categories*. Vol. 66 of *LI Monograph*, MIT Press, Cambridge, MA.

Plank, Frans (1995): (Re-)Introducing Suffixaufnahme. *In:* F. Plank, ed., *Double Case: Agreement by Suffixaufnahme*. Oxford University Press, New York, pp. 3–110.

Richards, Norvin (1997): What moves where when in which language?. PhD thesis, MIT, Cambridge, Massachusetts.

Richards, Norvin (2013): ‘Lardil “Case Stacking” and the Timing of Case Assignment’, *Syntax* **16**, 42–76.

Rizzi, Luigi (2007): ‘On Some Properties of Criterial Freezing’, *CISCL Working Papers on Language and Cognition* **1**, 145–158.

Rizzi, Luigi, ed. (2004): *The Structure of CP and IP. The Cartography of Syntactic Structures*. Oxford University Press, New York.

Ross, John (1967): Constraints on Variables in Syntax. PhD thesis, MIT, Cambridge, Mass.

Schweiger, Fritz (2000): ‘Compound Case Markers in Australian Languages’, *Oceanic Linguistics* **39**(2), 256–284.

Starke, Michal (2005): Nanosyntax class lectures. University of Tromsø.

Vainikka, Anne and Pauli Brattico (2014): ‘The Finnish Accusative: Long Distance Case Assignment Under Agreement’, *Linguistics* **52**, 73–124.