Nanosyntax: some key features

Pavel Caha*

Masarykova univerzita Brno

February 2, 2019

1 Introduction

Nanosyntax (Nano, Starke 2002, 2009; Caha 2009) is a theory of morphosyntax whose central tenets overlap to some extent with those proposed in Distributed Morphology (DM, Halle and Marantz 1993, 1994). For instance, DM's famous dictum of 'syntax all the way down' is something that Nanosyntax would subscribe to just as much as DM. Similarly, both theories converge on a 'late insertion' approach to morphology, where syntactic computation, consisting minimally of Merge and Move, applies before spell out. The shared features are given in (1).[1]

(1) Nanosyntax and DM, shared features:

a. *Late Insertion*: All syntactic nodes systematically lack all phonological features. The phonological features are supplied – after the syntax – by consulting the Vocabulary Items (lexical entries) in the postsyntactic lexicon.

b. *Syntax all the way down*: Terminal nodes are organized into hierarchical structures determined by the principles and operations of the syntax. Since terminal nodes correspond to units smaller than words, it follows that syntactic principles govern

*My work on this paper has been supported by the Czech Science Foundation, Grant no. GA17-10144S. I want to thank Karen De Clercq, Michal Starke and Guido Vanden Wyngaerd for their helpful comments on a previous version of this paper.

[1]However, I have to add that the shared features are somewhat compromised by the fact that I had to change the definitions compared to the wording used in Halle and Marantz (1994) (their literal rendering would make such an agreement impossible). Nevertheless, I hope to have extracted the spirit correctly in an attempt to show what both frameworks share, compared to other approaches, such as, e.g., A-morphous Morphology (see Anderson 1992) or Lexicalist approaches (e.g., Di Sciullo and Williams 1987).

> the internal structure of words. There is no sharp boundary between the traditional syntax and morphology.

At the same time, there are also differences. The first major difference is that in Nanosyntax, each morphosyntactic feature corresponds to its own syntactic terminal (cf. Kayne 2005, Cinque and Rizzi 2010). I will refer to this as "No Bundling." The second important difference is that there are no post-syntactic operations in Nanosyntax ("No Morphology," see, e.g., Koopman 2005, Kayne 2010)). In this article, I shall begin from No Bundling, and explain the idea behind "No Morphology" later on.

Let me start by noting that as a consequence of No Bundling, it follows that any complex object like, e.g., the set of features [1st PLURAL], must be created by (binary) Merge, and therefore, correspond to a syntactic phrase. This position reflects a more fundamental hypothesis about language, namely that any complex grouping of more primitive building blocks should be dealt with within a single generative system.

DM does not share this view. Ever since its early days until the most recent incarnations, the input to syntactic computation in DM are not only terminals with single features, but also complex sets of features, called feature bundles. As Bobaljik (2017) puts it in his recent overview of the theory, the input to the syntactic computation in DM is "a list of the syntactic atoms [...]. Items on this list would include [...] (possibly language-particular) bundles of features that constitute a single node: for example English (plausibly) groups both tense and agreement (person and number) under a single INFL node in the syntax." The reliance on 'syntactically atomic' feature bundles is characteristic for the work done in DM despite the fact that a lot of its home-grown research has been devoted to uncovering the rich structure of these bundles (cf. Harley and Ritter 2002). In fact, this type of work lays bare a fundamental tension inside the DM model: on the one hand, it is becoming increasingly clear that feature bundles have a rich internal structure. On the other hand, this structure cannot be generated by syntax, because it is already in place when syntax starts assembling such bundles together.

Feature bundles are a point of concern for Nano. As Starke (2014a) puts it, "a 'feature bundle' is equivalent to a constituent," because "enclosing elements inside square brackets is a notational variant of linking those elements under a single mother node." From this perspective, feature bundles are equivalent to n-ary trees, with n typically greater than 2. Starke further points out that this is equivalent to having a "second syntax" with "a new type of Merge for

the purpose of lexical storage." Crucially, the issue of what generative system lies behind the formation of these bundles remains a largely unaddressed question within DM.

Nanosyntax, as already mentioned, entirely dispenses with feature bundles. The framework (as a core hypothesis about the architecture of grammar) simply rejects the possibility that feature bundles may be generated outside of the core syntactic computation:

(2) *No Bundling (a property of Nanosyntax, but not DM):*
The atoms (terminal nodes) of syntactic trees are single features. All combinations of morphosyntactic features arise as the result of (binary) Merge. Pre-syntactic feature bundles do not exist, they correspond to phrases assembled by syntax.

The elimination of feature bundles eradicates the residue of Lexicalism in the theory of grammar. To see that, consider the fact that feature bundles are language specific (recall the quote from Bobaljik's 2017 introduction to DM). If that is so, then the list of "pre-syntactic building blocks" from which syntactic structures are constructed in DM is also necessarily language specific, as used to be the case in Lexicalist theories. DM of course differs from Lexicalist theories in many important respects, but the idea that syntax begins from language-particular objects is shared between DM and Lexicalism.

In Nanosyntax, this residue of a language-particular presyntactic lexicon is eliminated. What we are left with as the building blocks are universal features. Concerning the inventory of such features, Nanosyntax in line with Cartography adopts "the strongest position one could take; one which implies that if some language provides evidence for the existence of a particular functional head (and projection), then that head (and projection) must be present in every other language" (Cinque and Rizzi 2010, 55). As a result, the set of syntactic building blocks is the same in all languages, and we have no trace left of a language particular list that feeds syntax. Nanosyntax thus completes the shift from a presyntactic lexicon to a postsyntacic one.

The goal of this chapter is to further elaborate on the technical consequences of the differences highlighted above. I will introduce phrasal spellout and spellout-driven movement as the essential theoretical tools which allow the theory to capture a range of data while adhering to the principles introduced above. I will then briefly go through three case studies (on case marking, comparatives and root suppletion), which, I think, provide a good illustration of how the shared features and differences play out in the analysis of particular pieces of data.

2 Constituent spellout

Once feature bundles are dispensed with, it follows that markers like *we*, which corresponds to multiple features ([1 PL]), spell out multiple terminals of the syntactic tree (cf. Vanden Wyngaerd 2018). The first question is how to delimit the sets of terminals that may be spelled out by a single marker. This question arises because it is not the case that just about any two terminals may be pronounced together regardless of their position in the tree. To give an example: a morpheme like *we* cannot lexicalise the person of the subject and the number of a fronted XP, so that a sentence like *In god **we** trust* would be the spellout of a meaning corresponding to *In god[**-s I**] trust*. Here the bracket in the second sentence encloses two elements that jointly provide both the feature of the first person and that of a plural, yet their adjacency is not sufficient for joint spellout. So clearly, *some* restrictions on which features may be lexicalised together must be a part of the spellout mechanism: the sheer reduction of the number of morphemes in a string cannot be the right criterion.[2]

Starke (2002; 2009; 2014b; 2018) as well as much current work in Nanosyntax propose that the theoretically simplest way to define sets of terminals eligible for joint spellout is to rely on a grouping mechanism that is already needed for independent reasons. The sets of terminals that syntax provides for free are (by definition) constituents, and hence, the zero theory is one where morphemes spell out constituents (cf. McCawley 1968, Weerman and Evers-Vermeul 2002, Neeleman and Szendrői 2007, Radkevich 2010 for similar approaches outside of Nanosyntax).

Constructing theories along these lines has been the golden standard of work in generative grammar. Consider, for instance, the work done on ellipsis or movement. Here, we also encounter situations where ellipsis/movement targets multiple terminals. The standard way of explaining why several terminals undergo ellipsis/movement jointly is to say that they are all contained in a single constituent, and it is this constituent that actually undergoes ellipsis/movement. Nanosyntax adopts the same methodology (just applying it to spellout) and adheres to the view that when multiple terminals are joined inside a single morpheme, this is so because the morpheme spells out a constituent containing these terminals. All observable restrictions on joint lexicalisation should follow from this (in the same way as restrictions on 'joint movement' or 'joint ellipsis').

In following this logic, Nanosyntax not only adheres to a standard theory-building proce-

[2]Note incidentally that phenomena like these are not completely out of this world; see Blix (2016, sec.6.2) for a discussion of a morpheme of Pazar Laz, which is able to spell out the number of the object alongside the person of the subject, provided they form a unit available for spellout.

dure, but also increases our model of grammar incrementally (rather than changing it completely). To see that, consider the fact that where DM has feature bundles located under a terminal, Nanosyntax has a run-of-the-mill syntactic phrase. However, all terminal nodes of standard DM (where markers are inserted) are still syntactic nodes in Nano—just phrasal.

This is not so in sequence-based approaches to spellout. These approaches are close to Nano in that a single morpheme may correspond to several terminals. The difference is that in this theory, spellout targets multiple terminals that form "a functional/linear sequence" (Abels and Muriungi 2008, Dékány 2012, Svenonius 2012, Merchant 2015, Haugen and Siddiqi 2016). These approaches thus lead to a much more radical departure from feature constituency used in DM. In such approaches, the feature bundles of DM (targeted by insertion) are no longer constituents at all: they have been replaced by a new type of object, a "sequence."

2.1 Underspecification

Let me now move on to the observation that if we want to have a theory where morphemes target phrasal nodes, we must define our insertion principles differently from what is usually assumed in DM. The reason is that non-terminal insertion clashes with one of DM's key features (as defined in Halle and Marantz 1994), namely *Underspecification*, see (3).

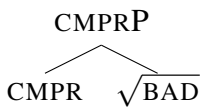
(3) *Underspecification (a feature specific to DM).* In order for a Vocabulary Item to be inserted in a terminal node, the identifying features of the Vocabulary Item must be a subset of the features at the terminal node.

Underspecification is embodied in the well-known insertion principle used in DM, the *Subset Principle*, given in an abbreviated form below. Notice that this principle explicitly states that it governs insertion only at terminal nodes.

(4) *The Subset Principle* (abbreviated, Halle 1997)
The phonological exponent of a Vocabulary Item is inserted into a morpheme of the terminal string if the item matches all or only a subset of the grammatical features specified in the terminal morpheme.

Can this insertion principle be broadened in a way that the Subset Principle could also govern insertion at phrasal nodes? It turns out it cannot. Consider, for instance, the suppletive com-

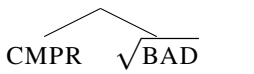
parative *worse*. Bobaljik (2012) proposes that it spells out a non-terminal composed of the root $\sqrt{\text{BAD}}$ and an associated CMPR head. Its lexical entry is as shown in (5):

(5) *worse* ↔ 

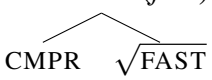
To allow for insertion of such lexical items, one could simply drop the restriction on terminal nodes from the Subset Principle. This would preserve the spirit of *Underspecification* and, at the same time, allow insertion into all nodes in general. To reflect the more general nature of insertion sites in such a reformulation of the Subset Principle, given in (6), I will call it the 'Generalised' Subset Principle. The boldfaced parts highlight the modifications introduced in (6) compared to the standard formulation in (4).

(6) *The Generalised Subset Principle* (a made-up principle that would fail, if proposed)
The phonological exponent of a Vocabulary Item is inserted into a **node** if the item matches all or only a subset of the grammatical features specified in the **node**.

This principle would allow the insertion of *worse* (5) into a non-terminal built by syntax, as shown in (7). The tree here depicts the structure built by syntax, and the bracket indicates that the spellout of CMPRP is successful, since *worse* in (6) matches a (trivial) subset of the features specified inside the CMPRP, namely the root $\sqrt{\text{BAD}}$ and the CMPR feature.

(7) 

However, the Generalised Subset Principle would also allow for the CMPRP in (8-a) to be pronounced by a regular (non-suppletive) root like *fast*, with an entry as given in (8-b). The spell out of the CMPRP in (8-a) is allowed because *fast* is specified for a (proper) subset of the features contained in the CMPRP node, specifically for the root $\sqrt{\text{FAST}}$. This is obviously a wrong result, since *fast* does not have a comparative meaning, and so insertion at CMPRP must be blocked in this case. In fact, taking this logic to its extreme, a whole sentence containing the root $\sqrt{\text{FAST}}$ at the bottom could be spelled out as *fast*.

(8) a. 

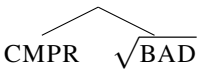
b. *fast* ↔ $\sqrt{\text{FAST}}$

There are various ways in which theories based on *Underspecification* may block *fast* in spelling out the whole CMPRP in (8-a). The main strategy is to augment the Subset Principle by additional principles that restrict *Underspecification* at non-terminals. For instance, Bobaljik adopts the Vocabulary Insertion Principle proposed by Radkevich (2010) for the case at hand; cf. Feature Portaging in Newell and Noonan (2018) or negative features in Siddiqi (2006). I shall not discuss these theories in any detail here (see Caha 2018a for the discussion of some potential problems), since the general point is exactly this: in order to allow for non-terminal spell out, one needs to re-think how insertion works: *Underspecification* on its own is not enough. Nanosyntax—rather than proposing additional principles on top of *Underspecification*—replaces *Underspecification* by a similar (but inverted) condition, namely *Overspecification*. Once *Overspecification* is adopted, phrasal lexicalisation works with no need for any additional principles.

2.2 Fusion and the architecture of grammar

In mainstream DM, however, insertion is proposed to target only terminals. Trivially, this move eliminates the need to augment *Underspecification* by any additional principles to deal phrasal spell out. However, the need to associate *worse* to multiple terminals then requires something special.[3] One possible approach within DM is to introduce a post-syntactic operation called Fusion. What Fusion does is that it takes the relevant non-terminal as an input, and turns it into a terminal, see (9).

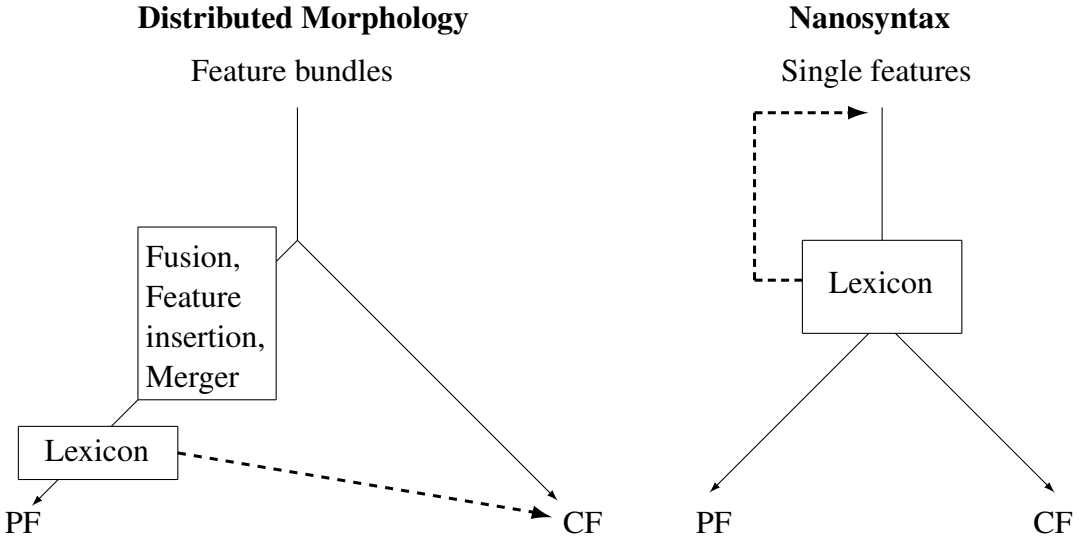
(9) Fusion: CMPRP $\rightarrow$ [CMPR $\sqrt{\text{BAD}}$]



The availability of such a proposal rests on a particular architecture of the grammar. In particular, because of the fact that Fusion does not affect the interpretation of CMPR, Fusion (along with other similar operations) is assumed to take place on a separate branch of the derivation that only affects PF. Still, Fusion happens before the insertion of Vocabulary items, which means that Vocabulary Items must also be inserted at the PF branch, following Fusion (and, as we shall see, other postsyntactic operations). The overall model is thus as shown in (10) on the left-hand side, which is a picture taken from Harley and Noyer 1999, slightly simplified).

[3] It is of course also possibe to deny that *worse* realises multiple terminals, in which case the CMPR marker would be silent. See Caha (2018a) for a discussion of the issues that interact with this decision, specifically, what kind of consequences this has for the so-called *ABA property of paradigms that we will turn to shortly.

(10) The architecture of Distributed Morphology (left) and Nanosyntax (right)



The point of Vocabulary-Item insertion is labelled as 'Lexicon.' The label is used because it is here where syntactic features are paired with their pronunciation. (In contrast, I will not use the term Lexicon for the pre-syntactic list of features or feature bundles, since the pre-syntactic list does not contain such pairs, at least in Harley and Noyer 1999.) In DM, the Lexicon is pushed down the PF branch, because, as said, it needs to follow Fusion, which does not feed interpretation. This, however, leads to a paradox. To see that, consider the fact that the CF needs to know whether *dog* or *cat* has been inserted into the root node. But since the nodes of the syntactic tree are devoid of phonological and conceptual features (recall (1-a)), this information is only present in the derivation after Vocabulary Insertion, i.e., after the 'Lexicon' box. So in order for the CF to know which Vocabulary Item has been inserted, it needs to have access to the stage of the derivation that follows Vocabulary Insertion. This is paradoxical, because the PF—CF split actually precedes Vocabulary Insertion, so if one sticks to the strict Y-model, CF should not be able to see which Vocabulary Item has been inserted. The tension is resolved (in Harley and Noyer 1999) by enriching the model by a direct communication line between the CF and the PF. This is indicated by the dashed line, which by-passes the syntactic derivation.[4]

Nano rejects post-syntactic operations, replacing Fusion by phrasal spellout. Therefore, the Lexicon is not located down on the PF branch, but appears at the juncture of the three systems (syntax, PF and CF). When a lexical item is inserted at a node, its phonology (e.g., *good*) is sent to PF, while the corresponding concept (**good'**) is simultaneously sent to CF. No additional direct communication line between the PF branch and the CF is needed. (The meaning of the

[4]Harley (2014) or Embick and Noyer (2007) provide a different solution to the issue of how conceptual information is passed on to CF (without the need for a direct communication line), which I discuss in section 4.2.

dashed arrow leading from the Lexicon back to syntax will become clear in section 4.1.)

Even though the existence of Fusion and similar operations complicates the architecture, mainstream DM has fully embraced the model with Lexicon on the PF branch, preceded by a number of operations, some of which are listed in (10). As Bobaljik (2017) puts it, in DM, "the investigation of mismatches between syntactically-motivated representations, and those observed in the morphophonological string" assumes "a central role," and "a variety of devices [=postsyntactic operations] serve together to constitute a theory of possible mismatches."

In Nano, the theoretical goal is different, namely to develop a unified theory of syntax and morphology based on No Bundling. On this approach, all such 'mismatches' must be accommodated by updating our syntax. Reference to post-syntactic operations represents a type of solution that would not be considered satisfactory in Nano.

Note that deciding whether we do—or don't—need a Morphological Component of the sort envisaged in DM is not a matter of simple empirical observation (as researchers working in DM sometimes tend to suggest). This is not to say that facts play no role, but it is an indisputable fact that for as long as the theory of syntax is not ready and finished once and for all (accounting for all the facts there are), mismatches between surface forms and syntactic structures are bound to occur: their very existence does not constitute evidence for anything. The question is rather how we proceed when mismatches are uncovered: do we treat them as illusions, which disappear once syntax is properly set up, or do we accommodate the facts by adding 'mismatch-removing operations' on top of an existing theory? Nanosyntax (along with other approaches) rejects the latter and opts for the former. To express the programmatic resistance of Nanosyntax to a post-syntactic component through a slogan, Caha (2007) suggested that one should not only avoid morphological analysis in the privacy of one's own Lexicon (as Marantz 1997 proposed), but also in the privacy of one's own Morphology.

(11) *No Morphology (a feature of Nanosyntax).* There is no component of grammar other than syntax that has the power to manipulate syntactic structures. No nodes or features may be added or deleted outside of syntax, no displacement operations take place outside of the syntactic computation.

As the chapter unfolds, I would like to give the reader a sense of the technology that Nano uses to deal with the scenarios that DM covers by the post-syntactic operations given in (10).

2.3 Overspecification

As the frst step towards a model without post-syntactic operations, Nanosyntax replaces all analyses with Fusion by phrasal lexicalisation. To avoid the problems caused by *Underspecification*, Starke (2009) replaces it by *Overspecification*, see (12).

(12) *Overspecification (a feature specific to Nanosyntax).* In order for a lexical item to be inserted in a node, the lexical entry must fully contain the syntactic node, including all its daughters, granddaughters, etc., all the way down to every single feature dominated by the node to be spelled out, and in exactly the right geometrical shape.

Technically, overspecification is implemented by the so-called Superset Principle:

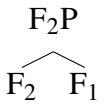
(13) *The Superset Principle, Starke (2009)*:
A lexically stored tree L matches a syntactic node S iff L contains the syntactic tree dominated by S as a subtree.

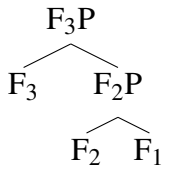
Once The Superset Principle is in place, non-terminal spell out works as needed (and without the need to add additional principles). Specifically, *worse* (recall (5)) can still spell out the structure [CMPR $\sqrt{\text{BAD}}$] in (7), because the syntactic tree is contained in the lexically stored tree in (5). However, *fast* (with the entry as in (8-b)) can no longer spell out [CMPR $\sqrt{\text{FAST}}$], because such a syntactic tree is not contained in the entry of *fast* in (8-b). So all is well.

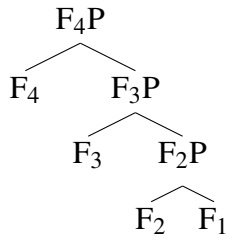
This is an important result. It shows that once *Overspecification* is adopted, we no longer need to search for additional principles that counteract the effects of *Underspecification*, and we have a fairly simple insertion rule that in principle applies to all nodes (both phrasal and terminal), an important achievement in the pursuit of No Bundling.

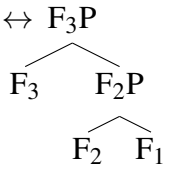
2.4 Elsewhere

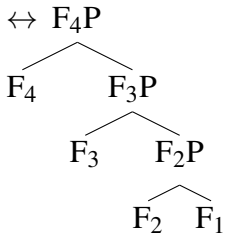
The Superset Principle (just like the Subset Principle) leads sometimes to the result that several candidate morphemes qualify for insertion at a particular phrasal node. Consider, for instance, the phrases given in (14) and the lexical entries as in (15).

(14) a. 

b. 

c. 

(15) a. 

b. 

What we see here is that the lexical entry for β matches all the structures in (14), because it contains every single one. α matches only (14-a,b), but it does not match (14-c). This means that for (14-a,b), both α and β are candidates for insertion. In such cases, the entries compete, and the so-called Elsewhere Condition (Kiparsky 1973) determines the winner. The Elsewhere Condition says that when two entries compete, the more specific entry wins. In our case, this is α, because it spells out proper subset of structures compared to β. As a rule of thumb, the more specific entry is the one that contains fewer superfluous features:

(16) *The Elsewhere Condition*:
When two entries can spell out a given node, the more specific entry wins. Under the Superset Principle governed insertion, the more specific entry is the one which has fewer unused features.

3 Features, Paradigms and the *ABA

With the basics of insertion in place, let me now turn to how the theory is put to use in modelling morphological paradigms. In a tradition going back at least to Jakobson (1936 [1962]), it has become customary to characterise the cells of paradigms in terms of more primitive units of analysis, namely features, where each cell of the paradigm is defined by a unique set of features. This approach is also widely adopted in DM, where the relevant features usually form a bundle at the relevant terminal. In Nanosyntax, feature bundles are dispensed with, and so each cell in a paradigm corresponds to a constituent containing the relevant features.

Consider, for instance, the trees in (14). These trees can be taken to define a paradigm like the one in (17). Specifically, the tree (14-a) contains the features F_1 and F_2; these corresponds

to the features that characterise the cell on the first row (Cell 1). The tree (14-b) corresponds to Cell 2 (it contains the same features as Cell 2), and (14-c) corresponds to Cell 3.

(17) An example paradigm

	features	α matches	β matches	insertion
Cell 1	F_1, F_2	yes	yes	α
Cell 2	F_1, F_2, F_3	yes	yes	α
Cell 3	F_1, F_2, F_3, F_4	no	yes	β

Consider now in addition that these cells (each cell representing a particular constituent) can be spelled out by lexical items which contain them. In (15), I have given two lexical entries such that α contains the features of Cell 2 and Cell 1, and β contains the features of all the cells. Such lexical entries therefore match the cells of the paradigms as depicted in the table (17). Where both match, competition arises with a winner determined by the Elsewhere Condition. The winners are recorded in the final column, and they correspond to the surface paradigm generated by the system introduced in the previous section.

The interest of "translating" the abstract structures in (14) onto a paradigm like (17) is that once we realise the possibility of such a "translation," we can start doing it also the other way round. If we succeed, we ultimately reduce surface paradigms (the sequences of αs and βs) to surface manifestations of syntactic structures of a rather familiar kind. A fundamental question in this enterprise is how we come to know—given a set of forms—what order they come in, so that we can then decide what structure they correspond to.

An important stepping stone on this path was the investigation of the so-called *ABA patterns. ABA (without the asterisk) refers to a pattern where in a particular arrangement of cells, the first cell and the last cell are the same, while the middle cell is different. When ABA is preceded by an asterisk, this means that such a syncretism is not found. As in any area of science, the goal is to explain why we observe some patterns of behaviour, but never other patterns; so if *ABA is observed in a paradigm, we want the theory to be able to explain this.

It can be shown that when the cells in a paradigm are ordered in terms of growing complexity (as in (17)), then the *ABA restriction falls out from the theory. Consider the reasoning: In order for Cell 3 in (17) to be spelled out as β, the entry of β must be as in (15-b) (it must contain all the features of Cell 3). Now by virtue of containing all the features of Cell 3, and because we are dealing with a paradigm of growing complexity (by assumption), β necessarily also contains all the features of Cell 2 and of Cell 1. If that is so, then β also automatically

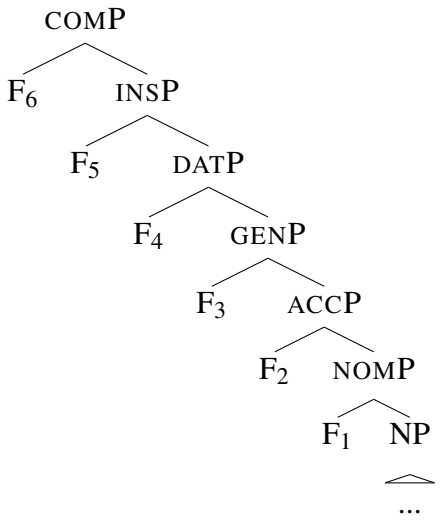
applies in those cells (in virtue of the Superset Principle).

Now since β is in principle applicable in all the cells, the only way how β may fail to spell out the middle cell is that there is a more specific competitor—α in our case—which outcompetes β due to the fact that it has fewer superfluous features. Crucially, once we have β in the most complex cell and α in the middle cell, we realize that the least complex cell (C1) can never be spelled out by β (which would yield an ABA type of pattern). To see that, consider the fact that in the setup we have created, both α and β can spell out Cell 1. Further, since α has fewer features than β, α it will always win when they compete, including Cell 1. The conclusion to be drawn here is therefore the following: if it is true that surface paradigms derive from syntactic structures of the sort in (14), we expect such paradigms to exhibit rather stringent restrictions on syncretism. For this reason, the study of *ABA patterns has been an important empirical domain to look at within Nanosyntax.[5]

In one of the first approaches along these lines, Caha (2009) has addressed this issue for case morphology, and found a number of languages where such constraints have been independently observed in the existing literature. Moreover, he argued that the results of such studies can be generalised into a universal linear restriction on syncretism in case, such that in the sequence NOM—ACC—GEN—DAT—INS—COM, only adjacent functions can be syncretic. Leaving the subsequent ramifications of this ordering aside (see Hardarson 2016, Starke 2017, Zompì 2017, Van Baal and Don 2018, Caha 2018b), Caha (2009) proposed that such a constraint can be explained by organising the cells of case paradigms in a cumulative fashion, as depicted abstractly in table (17), so that ultimately, the full case structure looks as given in (18).

[5]See, e.g., McCreight and Chvany (1991), Plank (1991) or Johnston (1996) for the investigation of *ABA patterns outside of both DM or Nanosyntax. See Bobaljik (2012) for an important discussion of *ABA within DM, which has provided much inspiration for this kind of work.

(18)



The proposal embodied in this structure is that the nominative case (corresponding to NOMP in (18)) is characterised by the feature F_1, the accusative (corresponding to ACCP in (18)) is derived from the nominative by yet another feature, etc. The proposal has the effect that the cases NOM—ACC—GEN etc. stand in a containment relation, exactly as the abstract structures in (14). The labels of the phrasal constituents are apparently exocentric, but this is only for clarity: the 'true' label of the nominative is F_1P, but this would be a rather opaque label, so the non-terminal nodes carry the name of the case, which is defined by the collection of Fs it dominates (e.g., ACC = [F_1, F_2]). I am leaving the content of the features aside, see Caha (2013) for some remarks.

The novel part of this proposal is not so much the idea that case decomposes into various features; this has been a common stance ever since Jakobson's (1936 [1962]) pioneering work. The novel part is rather that each case feature corresponds to a head of its own, which, recall, is one of the core features of Nano. What is also different from Jakobson is that the features are privative. In sum, the proposal derives the morphological patterns found in case paradigms from a type of architecture that is characteristic for syntactic derivations.

By now, a number of researchers working within Nanosyntax have looked at various phenomena through a similar lens, and used *ABA patterns as a tool for uncovering the underlying features and their hierarchical organisation into 'nesting' structures of the type in (18) (see in particular Starke 2009; Pantcheva 2010; De Clercq 2013; Baunaz and Lander 2018a,c; Lander and Haegeman 2018; Taraldsen Medová and Wiland 2018). And even though some more recent contributions point out that *ABA patterns can also be derived with non-nesting types of structures (e.g., Caha 2017a or Bobaljik and Sauerland 2018), this changes nothing to the fact that for a number of domains, the existence of *ABA patterns has led to the confirmation of an

architecture where the atoms of syntax are not feature bundles, but single features.

In DM, the standard treatment of case morphology is different in a way that I think is symptomatic for the larger architectural differences between the frameworks (see Halle 1997; Halle and Vaux 1998; McFadden 2004; Embick and Noyer 2007; Calabrese 2008). Consider, for instance, one specific proposal taken from Embick and Noyer (2007), a state-of-the-art paper on DM. Their feature decomposition, intended to capture the facts of the Latin declension, is given in (19). I have taken the freedom to re-label their ablative as instrumental, since in Latin, the ablative marks also instruments.

(19) Case decomposition in DM

	NOM	ACC	GEN	DAT	INS
Oblique	-	-	+	+	+
Structural	+	+	+	+	-
Superior	+	-	-	+	+

To see the generative power of such a decomposition, consider, for instance, the triplet NOM—ACC—GEN. In this sequence, no ABA pattern is attested in Latin, which is in line with the fact that this would be very rare crosslinguistically. In particular, Baerman et al. (2005) report that if one of NOM/ACC is the same as an oblique case (frequently a genitive), this is going to be the accusative and not the nominative.[6]

However, the decomposition in (19) cannot rule out such ABA patterns, and so it does not allow us to capture the asymmetry reported by Baerman et al. (2005) (cf. McFadden 2017, Smith et al. 2018). Consider the reasoning: In the proposal (19), the three cases under discussion share the feature [+structural], and so any exponent marked for [+structural] can appear in all the cases, yielding an AAA pattern (recall that DM uses *Underspecification*). When such a 'default' AAA pattern interacts with competing entries, ABA patterns emerge. Specifically, because of the decomposition into equipollent features, it is possible to devise tailor-made competitors for each individual case. Suppose, for instance, that NOM has a dedicated case marker, B, specified as [-Oblique, +Structural, +Superior]. Its competition with the underspecified marker A would yield a BAA pattern. However, if B were tailor-made for ACC, competition would yield ABA, and if B were tailor-made for GEN, we would get AAB. This shows that within this particular triplet, any pair of cases can be syncretic, which goes against the general-

[6]Baerman et al. (2005) phrase this as a tendency, see Caha (2018b) for a discussion of some counterexamples.

isation observed in the typological literature. Hence, as Caha (2009) argues, the Nanosyntactic proposal that features are privative, and assembled by Merge, is not only theoretically attractive (consistent with No Bundling), it allows one to capture important generalisations.

The standard DM account (as described in Embick and Noyer 2007) has an additional feature that is worth mentioning in this context. Following Marantz (1991) and McFadden (2004), Embick and Noyer (2007) report that in DM, case features are not a part of the syntactic derivation at all. They note: "At PF, case features are added to DPs [...], based on the syntactic structure that the DP appears in. [...] These features are added at PF, and are not present in the syntactic derivation." I will refer to such features introduced post syntax as morphological features, 'M-features' for short.

The important point is that the postulation of M-features amounts to the introduction of yet another generative component (this time post-syntax), where complex feature bundles can be assembled. In addition, when case is expressed independently of other categories (as in agglutinative languages), such case features would be introduced in a separate node, also created post-syntax, as Embick and Noyer (2007) make clear in their footnote 25. I find it difficult to reconcile such an array of structure-building operations with the explicit statement that in DM, "all complex objects, whether words and phrases, are treated as the output of the same generative system (the syntax)" (Embick and Noyer 2007). One can of course always backtrack from specific proposals about case features, but the fact remains that from the perspective of Nanosyntax, the multitude of mechanisms that DM's architecture makes available is "an embarassment of the riches," as Bobaljik (2017) points out.

To make explicit the implications of these findings for the general architecture assumed in Nano and DM, I will start by quoting a passage from Embick and Noyer (2007), originally meant as a guideline for comparing Lexicalist and non-Lexicalist approaches. They say: "It is often objected in discussions of non-Lexicalist versus Lexicalist analyses that the patterns analyzed syntactically in the former type of approach *could* potentially be stated in a theory with a Lexicon. This point is almost certainly correct, but at the same time never at issue. [...] The Lexicalist position, which posits two distinct generative systems in the grammar, can be supported only to the extent that there is clear evidence that Lexical derivations and syntactic derivations must be distinct."

DM (compared to Nano) has exactly the same issue of multiple systems that can generate (or minimally provide) complex objects: (i) pre-syntactic feature bundles, (ii) syntax, (iii) feature

bundles constructed at 'PF,' (iv) nodes inserted at PF. So if the reasoning quoted above is followed consistently, it must be concluded that Nano has an architectural advantage of exactly the same sort that differentiates between lexicalist and non-Lexicalist approaches. In particular, to the extent that feature bundles can be generated by syntax (corresponding to vanilla-flavour syntactic constituents), they should not be drawn from a pre-syntactic list or created at PF.

4 Cyclic spellout

Another important feature of current work in Nanosyntax is *Cyclic spellout* (cf. Starke 2018, Baunaz and Lander 2018b, Caha et al. 2019a).

(20) *Cyclic spellout*. Spell out must successfully apply to the output of every Merge F operation. After successful spellout, the derivation may terminate, or proceed to another round of Merge F, in which case a new round of spellout is initiated, and so on.

Cyclic spellout plays a central role in current Nanosyntactic thinking, providing the basis of an account for a number of phenomena including idioms, root suppletion and affix ordering. To see how cyclic spellout works, let us assume the very same toy scenario that we have been working with in (14), only enriched by the idea of cyclic spellout. Suppose then that syntax merges F_1 and F_2, forming F_2P:

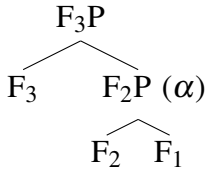
(21) $[_{F2P}\ F_2\ F_1\ ]$

After Merge F has applied, spellout applies. Spellout means that the lexicon is searched for an item matching the phrase in (21). In our toy scenario, F_2P is contained in the lexical entry for both α, recall (15-a), and β, recall (15-b). Recall also that α wins against β due to The Elsewhere Condition. Spellout is therefore successful.

After the successful application of spellout at F_2P, the lexicalisation procedure remembers minimally that F_2P can be lexicalised by α. If no more features are added (we are finished constructing the intended meaning), the derivation terminates and F_2P will ultimately be pronounced as α. However, if we want to add more meaning, the derivation continues—without being immediately pronounced. Suppose it continues, and that F_2P is fed back to syntax for an additional Merge F operation. The result is that F_3 is added, and spellout applies to the F_3P depicted in (22). In (22), the tree contains the information (accessible to the lexicalisation

procedure, not to syntax) that F_2P has been matched by α at the previous round of spellout.

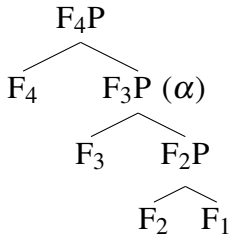
(22)



When a tree like (22) is fed to spellout, we again find two possible matches for F_3P, namely α and β, with α again as the winner. The match of α at F_3P is remembered, and all previous matches inside F_3P are forgotten (over-ridden). Should no more features be added, F_3P would be pronounced the same as F_2P, namely by α.

If we want to add more meaning, (22) is fed back to syntax again, and F_4 is added, producing F_4P as shown in (23).

(23)



Once again, at spellout, the product of Merge F (namely F_4P) must be matched against a lexical entry. This time, only the lexical entry for β is a match. It is thus remembered as the spellout of F_4P, and α is over-ridden. If no more features are added, F_4P is pronounced as β.

Note that lexical entries containing trees do not duplicate syntax in any way (which is similar to saying that lexical entries containing phonology do not "duplicate" phonology). The purpose of the lexicon in Nanosyntax is to link syntactic representations (trees) to representations legible by phonology (sound) and by the conceptual system (meaning).

Note finally that cyclicity here is neither the same as the notion of a phase as currently entertained in the syntactic literature (e.g., Chomsky 2001), nor is it meant as its replacement. The core of Chomsky's proposal is that some phrasal nodes are special and correspond to phases, other phrasal nodes are ordinary, and do not correspond to phases. Cyclic spellout treats all phrasal nodes alike. Phases in Chomsky's sense are not a part of the standard Nano toolbox, but they could be easily added; there is no logical incompatibility between cyclic spellout (cyclic lexical look up) and the idea that some phrasal nodes are special (for instance, where actual shipping to PF/CF occurs).

There are two empirical domains where cyclic spellout plays an important role. I now visit

them in turn.

4.1 Spellout-driven movement

When spellout at a newly formed FP fails, spellout-driven movements take place. The goal of these movements is to create a configuration where the spellout of FP succeeds.

(24) *Spellout-driven movement.* When Merge F produces an FP that cannot be spelled out (no lexical item matches the FP), the FP is rejected at the interface. Syntax then tries to rescue the structure by performing one in a predefined hierarchy of movement operations, before it sends the structure for spellout again. A variety of spellout-driven-movement operations may apply until spellout at FP succeeds. Once lexicalisation succeeds, the derivation either terminates or continues by Merge F.

Spellout movements are different from standard feature driven movements in that they have no effect on interpretation, they are strictly local (inverting the order of two adjacent phrases), and show no reconstruction effects (there is no evidence for two interpretation sites). Within Nanosyntax, spellout-driven movement is used as a replacement for the traditional head movement as well as for DM's Merger. The machinery is also related to the U20 type of movements proposed in Cinque (2005). However, wh-movement, focus movement, etc. are of a different kind and contrast with spellout-driven movement on all three properties given above (they affect interpretation, they can cross multiple phrases, they show reconstruction effects).

The algorithm for spellout-driven movement is given in (25). It is basically a version of the algorithm as presented in Starke (2018), and I will explain its workings step by step.

(25) Spellout Algorithm

- a. Merge F and spell out.
  - (i) If (a) fails, try spec-to-spec movement of the node inserted at the previous cycle, and spell out.
  - (ii) If (a.i) fails, move the complement of F, and spell out.
- b. If (a.ii) fails, remove F from the main workspace. Start a new workspace, and build a phrase containing F that can be spelled out. Once done, Merge that phrase back with the main projection line.

In the previous section, we have already informally talked about how (25-a) works. What we shall now see in more detail is what happens when spellout fails. I am going to ilustrate the algorithm on a fragment of data discussed in Caha et al. (2019a) (CDV henceforth). Once the system is introduced, I show that it is capable to replicate derivations that have been treated by post-syntactic Merger in DM.

The goal of CDV's paper is to capture alternations in comparative marking in Czech, English and other languages. Beginning with Czech, the basic contrast is illustrated in (26).

(26)

FULL MARKING			REDUCED MARKING		
POS	CMPR	GLOSS	POS	CMPR	GLOSS
chab-ý	chab-**ějš**-í	'weak'	slab-ý	slab-**š**-í	'weak'
kulat-ý	kulat-**ějš**-í	'round'	bohat-ý	bohat-**š**-í	'rich'
jist-ý	jist-**ějš**-í	'certain'	tlust-ý	tlust-**š**-í	'fat'

The table shows two different classes of comparatives in Czech. The first class—which is the productive one—can be seen on the left, and it forms comparatives by the suffix *-ějš*. The marker appears in a position preceding the final agreement marker *-í* (obligatory in Czech). The second class uses a reduced marker to the same effect, namely *-š*. There does not seem to be any straightforward way of deciding which adjective forms which type of comparative; this seems to be to a large extent an arbitrary property of a particular root (though frequency and phonology play a role, see Křivan 2012). In what follows, I will use the examples on the first row (both meaning 'weak') to illustrate the working of the theory.

CDV give various reasons to believe that the non-reduced marker *-ějš* decomposes into *-ěj* and *-š*, where the latter marker is shared between the two comparatives. For instance, the comparative adjective *chab-ěj-š-í* 'weaker' has a corresponding comparative adverb *chab-ěj-i*, which lacks the *-š*, suggesting that *-š* has an independent life on its own. The two classes thus differ as shown below, with the final agreement omitted:

(27) Two classes of comparatives

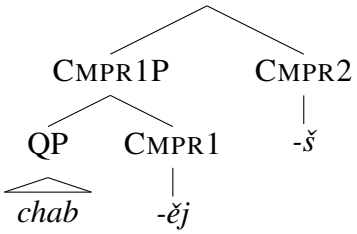
a. √ -ěj -š

b. √ -š

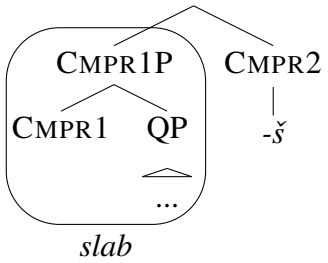
Taking the bi-morphemic nature of the comparative in (27-a) as a starting point, CDV propose

that in the morphosyntactic structure, two comparative projections must be present, where the lower one is pronounced as *-ěj*, and the higher one as *-š* (cf. Caha 2017b, De Clercq and Vanden Wyngaerd 2017). This is schematically depicted in (28-a), where the comparative markers appear on top of a QP. QP corresponds to a gradable adjective, and decomposes into the gradability head Q and the property head A (not shown in the tree).

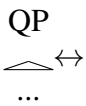
(28) a.

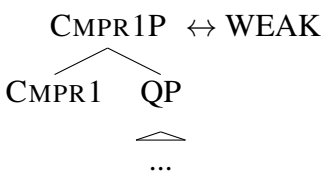


b.



In this setting, (28-b) encodes the proposal that Class 2 adjectives lack CMPR1 *-ěj* because their roots spell out a phrasal projection that includes CMPR1 as well as the QP. The structures are simplified, and I elaborate on them further below. However, what can be seen right away is that the two classes of roots can be easily distinguished in the lexicon as follows:

(29) a. *chab* ↔  ↔ WEAK

b. *slab-* ↔  ↔ WEAK

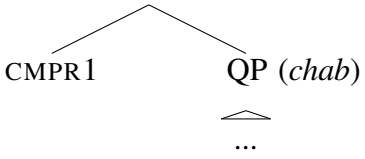
With such entries, both roots can spell out the QP (due to the Superset Principle), and appear as such (without any affixes) in the positive degree (which corresponds precisely to the QP). In the comparative, a difference shows up. The *chab* root, given in (29-a), still spells out QP only, and needs additional affixes to express CMPR1 (*-ěj*) and CMPR2 (*-š*). The *slab* root, however, is able to spell out CMPR1 on its own, and combines only with CMPR2 *-š*. It is interesting to note that this way, we state the selection requirements between the root and the particular comparative suffix using the variable size of the lexical tree associated with the root. There is no need for a statement of the sort 'this root combines with *-š*' or 'this root combines with *-ěj-š*;' such combinatorial statements simply fall out from the lexical difference in the size of the tree associated to the two classes of roots.[7]

Let me now describe how exactly spellout works in Class 1 (non-reduced marking). The derivation begins by forming a QP. Such a QP is contained in both lexical entries in (29), and

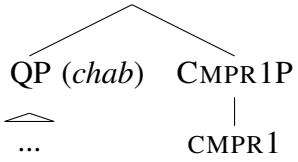
[7]This is an interesting proposal for allomorphy in general, and there is an ongoing work that investigates this option (see, e.g., Holaj 2018), but I leave this aside here for reasons of space.

so both roots can be inserted. I am assuming here (following CDV) that the choice of the root is free, and not subject to Elsewhere reasoning (cf. Harley and Noyer 1999). Suppose that the root in (29-a) (*chab*) is selected. QP is thus successfully spelled out, and the derivation continues by adding CMPR1, forming CMPR1P. When this happens, the structure is again sent for spellout. The structure now looks as in (30-a):

(30) a.



b.



(30-a) cannot be spelled out by the root *chab* (its lexical entry does not contain CMPR1), and so the output of MergeF (CMPR1P) ends up without a spellout. The structure is therefore rejected at the interface, and Merge F cannot continue. A repair spellout-driven movement is therefore attempted. The various options of this movement are always applied in the succession given in (25) (no look-ahead as to whether a particular step will succeed or not).

The first option is moving the Spec of the complement. For our case, this entails that the movement of Spec,QP should be tried first, but QP has no movable Spec in (30-a), so this option is skipped. The next option down the list is the movement of the full complement, which is the QP. The output of such a movement is shown in (30-b). Note that QP leaves no trace inside CMPR1P. According to Starke (2018), this is a general property of spellout-driven movement, and this is also how it differs from, e.g., *wh*-movement (recall that spellout-driven movement, unlike *wh*-movement, never shows any reconstruction effects, and so there is never any evidence for two different interpretive positions).

After movement, the spell out of CMPR1P is tried again. CMPR1P now lacks the QP inside, and so there is a chance that lexicalisation succeeds. We know that it does in Czech, inserting the marker *-ěj*. Its lexical entry according to CDV is therefore as shown in (31). It is easy to see that this entry perfectly matches the CMPR1P in (30-b). (Recall that it is the lower CMPR1P that has been created by Merge F, and it is therefore this lower node that undergoes spellout.)

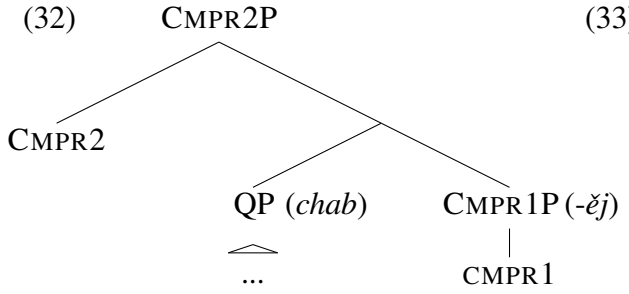
(31) *-ěj* ↔ CMPR1P
|
CMPR1

At this stage, the structure (30-b) is successfully spelled out, and if no more features are added, it would be pronounced as the sequence of *chab* and *-ěj*. Note that as a result of spellout-driven

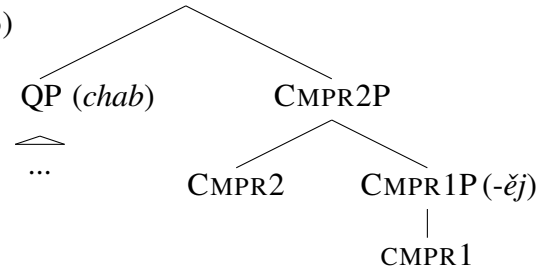
movement, CMPR1P follows the QP, so on the surface, *-ěj* follows *chab*. The suffixal nature of *-ěj* is determined by the shape of the lexical tree it is associated to in (31). The tree has just a single feature dependent on the lowest phrasal projection CMPR1P. In a model like that of Chomsky (1994) (Bare Phrase Structure), such a configuration only arises in syntax when the second daughter of CMPR1P extracts. And since movement is only to the left (as in Kayne 1994 and many others), this means that *-ěj* will only ever be inserted as a suffix.

The derivation now proceeds by merging CMPR2 on top of (30-b), with the result shown in (32). This constituent cannot be spelled out as is, triggering spellout-driven movement. According to the spell out algorithm, the first operation that must be tried is the movement of the Spec of CMPR2's complement. This phrase corresponds to the QP, and so the QP is moved out, with the result in (33). In Czech, there is no marker to spell the CMPR2P thus formed (containing the features CMPR1 and CMPR2), and hence, spell out fails.

(32)

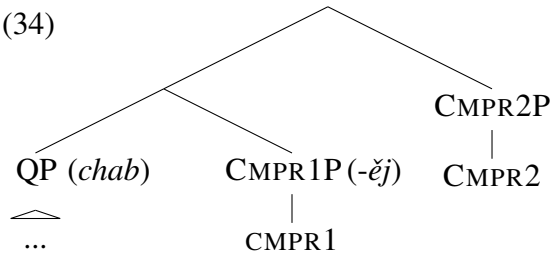


(33)



When (33) is rejected, the next option to be tried is complement movement. We start from the original Merge F structure (32), the complement of the newly added F is moved, producing (34). CDV propose that the lexical entry for *-š* is as in (35). This lexical item matches the CMPR2 in (34) out of which the phrase [*chab-ěj*] had extracted, and so spell out succeeds, producing the correct sequence of morphemes *chab-ěj-š*.

(34)



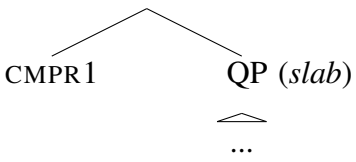
(35) *-š* ↔ CMPR2P
CMPR2

Note that this way, Nanosyntax replicates a roll-up movement type of derivation without the need to postulate the usual 'movement' features on particular heads. In this theory, mirror

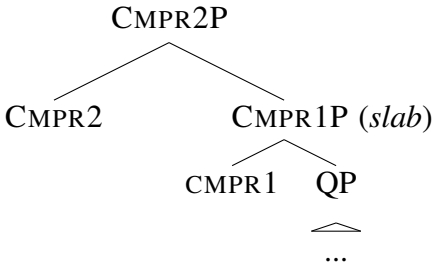
image orders (in the sense of Baker 1985) arise as a result of the interaction between the spellout algorithm and the tree shape of lexical entries (Starke 2014b).

I will now briefly show how reduced comparative marking arises in this theory. The derivation starts again by assembling a QP, which can be spelled out by *slab*, because it is contained in its entry, recall (29-b). If we wanted to produce the positive, the derivation would end here, producing just *slab* to which AGR would be added. In the comparative, when CMPR1 is added on top of such an QP, see (36-a), spell out succeeds without any movement, because exactly such a CMPR1P is contained in the lexical entry of *slab* in (29-b). CMPR1P is thus successfully spelled out, and the derivation continues by merging CMPR2 on top, see (36-b).

(36) a.

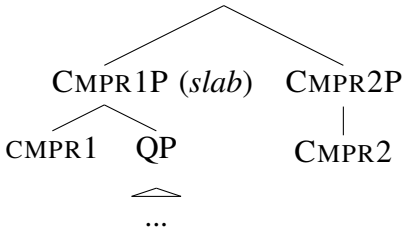


b.



This time, spell out fails, and spellout-driven movement is triggered. The first thing to be be tried is Spec movement. However, the complement of CMPR2 in (36-b) has no Spec, and so this option is skipped. Complement movement is tried next, producing the structure (37), which correctly spells out as *slab* followed by *-š*:

(37)



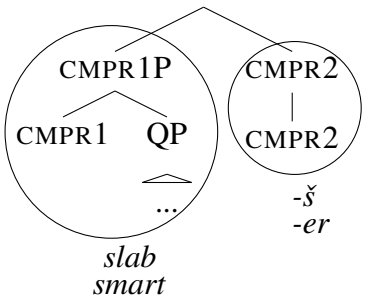
An important observation is that for different roots, the spellout algorithm produces different tree shapes, compare (37) with (34). The choice of a particular root thus has a certain (limited) power to steer the derivation in a particular direction. For example, the lexical item does not influence the sequence in which features are merged, but it does influence in how the features are linearly ordered (CMPR1 either precedes or follows the complement).

This turns out to be useful in extending this theory to English, focussing on the alternation between ***more*** *intelligent* and *smart-**er***. CDV build their analysis around the fact that the two markers differ in terms of complexity. In particular, *-er* is simpler than *mo-re* (it spells out fewer features). The complexity of *more* can be interpreted either literally, so that *more* is

segmented as *mo-re*. However, even in the absence of surface decomposition, the interpretation tells us that *more* is the comparative of *much*, which in Nano necessarily means that *more* (which is minimally [CMPR *much*]) must express more features than *-er* (CMPR). Importantly, each feature must correpond to a syntactic head.

CDV implement these observations as follows. First of all, they interpret adjectives like *smart-er* as exactly parallel to the Czech reduced comparatives like *slab-š* 'weaker,' with the final structures as in (38), where spellout-driven movement has moved CMPR1P out of CMPR2P in a way described for (37). Because of this parallel, the lexical entry of *smart* will be like the one of *slab* in (29-b), i.e., associated to the full CMPR1P.

(38)



Now we know that adjectives like *intelligent* do not combine with *-er*. CDV encode this by associating such adjectives to a QP only, see (39). This yields **intelligent-er*, since the two pieces (*intelligent* and *-er*) do not spell out all the features of the comparative (they lack CMPR1).

(39) *intelligent* ↔ QP
...

(40) *more* ↔ CMPR2P
CMPR2 CMPR1P
CMPR1 Q

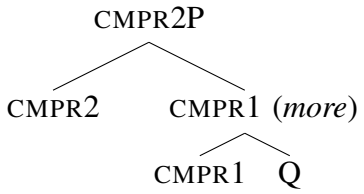
The lexical entry for *more* (which does appear in the comparative of *intelligent*) therefore needs to spell out both CMPR1 and CMPR2. This is independently supported by the fact that *more* on its own is the comparative of *much*, so it must be able to pronounce all the ingredients of the comparative meaning—plus the meaning of *much*. *Much* in CDV's system corresponds to a functional adjective spelling out Q, which lacks any particular property A associated to it. Therefore, if *more* is the comparative of *much*, its lexical entry must be as in (40).

With the lexical entries in place, let us see how the derivation proceeds. Syntax first assembles the QP and spell out finds a match, the item *intelligent*. The structure is then fed back to syntax for additional Merge F. CMPR1 is added, but since *intelligent* cannot spell out such a phrase, repair movements take place. However, neither Spec movement or complement move-

ment produce the right configuration for the spell out of CMPR1 by *more*. The last option (25-b) in the spell out algorithm is therefore followed. What the definition says is that we should remove F (CMPR1 in our case) from the main derivation, and open a separate derivational space, where we Merge CMPR1 with something else. This separate derivation is treated similarly to the main derivation, and the spellout algorithm works as usual. CDV propose that the first thing which happens when a new derivational space opens is that CMPR (which corresponds to F) is first attempted Merged with Q, which corresponds to F_{-1}. The result is shown in (41-a), which spells out as *more*, because it is contained in the lexical tree of *more* in (40).

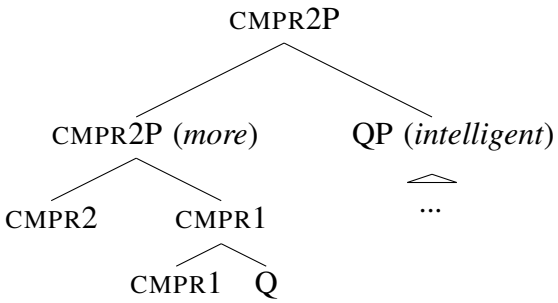
(41) a.

b.



The derivation of the comparative then continues in the new workspace, adding the next F to CMPR1P, see (41-b). This structure also spells out as *more*, and since the intended meaning has been assembled, the second workspace is closed (merged with the QP *intelligent*), with the result as shown in (42):

(42)



What we see here is that *more* is merged as a pre-modifier of *intelligent*, which is a consequence of the geometrical shape of its lexical tree. In particular, the lexical tree of *more* has two sister features as its lowest elements. This contrasts with suffixal markers, which only have a single feature as the bottom-most element. In Nano, the difference between prefixes and suffixes is thus encoded by the difference in the shape of the lexical tree.[8]

Note further that Spec formation has the effect that CMPR1 and CMPR2 form a constituent in (42) to the exclusion of the QP spelled out by *intelligent*. In DM, such a 'rebracketing' (from the scope order [CMPR1 [CMPR2 QP]] to [[CMPR1 CMPR2] QP]) is often treated as a

[8]Cf. Taraldsen et al. (2018); De Clercq and Vanden Wyngaerd (2018) for an analysis of prefixes as complex specifiers.

result of post-syntactic Merger. In Nanosyntax, this type of re-bracketing arises as a core part of the syntactic computation, executed by the Spellout algorithm (25). In particular, when Spec formation is used during a derivation, Merger type of structures are going to arise.

Let me now compare this analysis with how comparatives are treated in DM. For instance, in Matushansky (2013), *more* is the spell out of the very same syntactic Deg head that is spelled out by *-er*. The difference between the two spell out shapes is attributed to a post-syntactic *much* insertion, where a special node is created in the morphology that has not been present in syntax and adjoined to Deg. Into this M-node, *much* is inserted as a last resort support for the affixal *-er* in the Deg head.

Such an analysis does justice to the complex nature of *more*, but accounts for it by a post-syntactic operation, something that CDV manage to avoid. Usually, in DM, justification for M-nodes is provided by the observation that they do not contribute to interpretation. In this particular case, the *much* part of the comparative *more* is apparently not needed for interpretation, since adjectives like *smart-er* do without it. However, such an observation does not, on its own, require the existence of post-syntactic operations. In CDV's analysis, the superfluous nature of *much* is also captured, specifically by the fact that Q is present twice in the structure, once as a part of *intelligent*, and second, as a part of *more*. In the latter case, it is introduced simply as a way for CMPR1 to merge with something, so that the derivation can continue; hence, the analogy to Matushansky's *much*-support is clear (cf. Corver 1997 for the original idea). The crucial difference is thus not so much about the analytic intuitions here (Matushansky's paper is rather similar in spirit to the CDV analysis, arguing for syntactic treatment of comparatives); the difference is that the Nanosyntactic technology allows one to implement such intuitions without the need to postulate a second generative system with the power to add heads after syntax (duplicating Merge in Morphology).

In Bobaljik's (2012) treatment, there is no *much*-support. Both *more* and *-er* spell out exactly the same head. When this head is suffixal, *-er* surfaces. When it is not, *more* is inserted. I shall leave it aside whether (and how) this proposal expresses the fact that *more* is the comparative of *much*, and focus rather on the fact that the Morphological component is implicated also in Bobaljik's analysis. In order to govern the alternation between *more intelligent* and *smart-er*, he introduces the feature [+M] on roots that have the affix *er*. This feature triggers the application of a post-syntactic Merger operation, which (after syntax) attaches the CMPR head as a suffix to the adjective. One of the reasons for using post-syntactic Merger is the fact that the

two different realisations of the comparative are sensitive to the particular choice of a lexical item, which—in standard theories—should not influence how syntactic derivations proceed.

Clearly, the present analysis follows a very similar intuition, namely that there is a lexical difference between the two classes of adjectives such that some combine with *more* and some with *-er*. So again, the issue is rather *how* this intuition ends up encoded. In CDV, the difference between the two classes of roots is expressed in terms of how many meaning components the lexical items pronounce, an indispensable type of lexical variation. Via the cyclic-spellout algorithm (which ensures that the choice of a lexical item can influence, to a limited extent, how syntactic derivation proceeds), the difference in the size of the lexical items is reflected by different structures, as syntax tries to merge the comparative features on top of two different bases. Crucially, no reference to features like "I trigger post-syntactic merger" is needed, so any post-syntactic component that effects such instructions is not needed either. However, this is achieved not by dismissing the idea that the realisation of the comparative in English is a consequence of a particular lexical choice. This idea is fully embraced, but embedded into a model of syntax where such an observation can be implemented without invoking post-syntactic operations.

Finally, on Embick's (2007) analysis, it is left open as to whether *more* requires the addition of an M-head (as in Matushansky's analysis) or not (as in Bobaljik's analysis). But similarly to Bobaljik, he uses a post-syntactic operation of Local Dislocation to bring the affix and the adjectival root together. The difference between Local Dislocation and Merger is too subtle to be addressed here (see Matushansky 2013 for a good discussion of the problems associated to this analysis), and similar remarks thus apply as in the case of Merger.

4.2 Pointers: idioms and root suppletion

The cyclic nature of spellout further opens up the interesting possibility that insertion at higher nodes can make reference to lexical items inserted at an earlier cycle. A device that makes reference to a previous cycle is called a pointer in Nanosyntax (Starke 2014b, Caha and Pantcheva 2012, Taraldsen 2012, Vanden Wyngaerd 2018, Caha et al. 2019b).

(43) *Pointers.* Insertion at a cycle *n* can make reference to lexical items inserted at cycles preceding *n*. A device that allows for such a reference is called a pointer.

At least since Bobaljik (2000), a rather similar idea has been also a part of DM. The crucial difference between DM and Nano is how reference to lexical entries is executed. In DM, this is relegated to traditional contextual rules: they scan the vicinity of the insertion site, and if they find what they are looking for, a special allomorph may be inserted. However, such contextual rules may never change the lexical item they refer to. In Nanosyntax, this is different. In order to see it, consider the basic shape of an entry with a pointer (as used in Caha et al. 2019b). Such a lexical item looks as in (44):

(44) $\gamma \leftrightarrow$ FP

F LEXICAL ITEM 153

Such a lexical item can be used to spell out a node that contains the feature F as one of the daughters, and the other daughter corresponds to a phrase that has been spelled out by a particular lexical item (namely 153) at a previous cycle. Concerning the use of the numerical index: lexical items in general correspond to a particular triplet of phonology, syntax and concept (where phonology or concept may be empty). For ease of reference, we can assign a numerical index to each such a triplet. The specific lexical item which is referenced in (44) is thus the one that has the (arbitrarily assigned) index 153.

Now if at the stage F_{-1} the lexical item 153 is indeed used to spell out the structure, the spellout procedure remembers it, and goes for another round of Merge F. This leads to the addition of F, forming an FP as shown in (45). In the bracket, we once again find the information that $F_{-1}P$ has been spelled out in a particular way, namely by the lexical item 153. (Recall that this is a type of information that only spell out has access to, not syntax.)

(45) F_xP

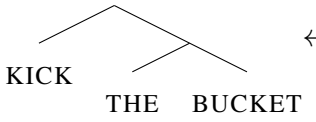
F_x $F_{-1}P\,(153)$

...

If the derivation reaches a stage like the one in (45), then the the lexical item (44) will be used to spell out such an F_xP. This will lead to the over-riding of the lexical item 153, and its replacement by the new lexical item, namely (44).

What kind of data do we need pointers for? The initial reason for Starke to introduce pointers was the need to deal with idioms like *kick the bucket*. The observation that Starke

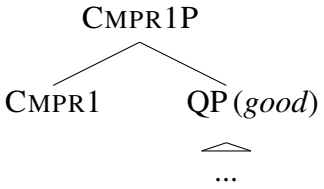
wanted to capture is that the idiomatic reading is only present in the context of specific lexical items contained in the idiom; so even though *pail* can refer to the same object as *bucket*, *kick the pail* lacks the idiomatic reading, which *kick the bucket* has. To encode the observation, Starke proposed that idioms are phrasal lexical items which make reference to particular lexical items, as in (46). Here small caps stand in for the lexical items that the spellout procedure had inserted at previous cycles.

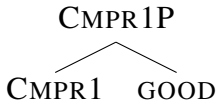
(46) 

↔ 'DIE'

When such a lexical item is used, it inserts the concept 'die,' over-riding the concepts associated to *kick* and *bucket*. Note that the entry is not associated to any phonology, so the phonology of the relevant entries remains unchanged; they sound exactly like *kick*, *the* and *bucket*.

The same technology can be used to deal with root suppletion. Suppose, for instance, that we insert the adjective *good* as the spellout of a QP. Suppose that a CMPR1 head is further merged with this structure, leading to the configuration as in (47-a). This structure is spelled out by the lexical item *bett* in English, which is given in (47-b), over-riding the previous spell out *good*. Note that the conceptual information is not over-ridden this time; only the phonology is. From this perspective, suppletion is nothing else but the phonological counterpart of idiomaticity.

(47) a. 

b. *bett* ↔ 

As Caha et al. (2019b) point out, this treatment of root suppletion shows some interesting differences to how root suppletion is treated in DM. DM has from the start assumed that roots are a-categorial entities, notated as $\sqrt{}$. Further, as a zero hypothesis, it was assumed that there is just a single $\sqrt{}$ in syntax, and that morphological exponents are inserted late into $\sqrt{}$ nodes (i.e., post syntax). This is important for modularity, which requires that only such information is present in a module, which the module can " read." Now DM is well aware of the observation that "[n]o phonological properties of roots interact with the principles or computations of syntax nor do idiosyncratic Encyclopaedic facts about roots show any such interactions" (Marantz 1995). In an architecture where there is a single $\sqrt{}$ in syntax, and various morphological roots

are inserted late, it follows that syntactic computation will not be able to make reference to phonological or conceptual information.

The work done in Nanosyntax on the issue of roots concurs with modularity as well, but differs in one respect. In particular, as Ramchand (2008) proposes, a-categorial $\sqrt{}$ nodes could easily be eliminated from syntax all together, and replaced by (sequences of) functional heads. But the main point is the issue how root suppletion interacts with late insertion of root morphemes, which turns out to be problematic in DM, but not in Nanosyntax. To see this, consider the fact that among the adjectival roots (*smart, fast*, etc.) we also find suppletive ones like *good*, see (48-a). Suppose further that lexical insertion targets only terminals, as in classical DM. What would be the entry for *bett*? The answer is that *bett* would be the contextual allomorph of the $\sqrt{}$ node in the context of a CMPR head, as in (48-b).

(48) a. $\sqrt{}$ ⇔ *smart, fast, good*

b. $\sqrt{}$ ⇔ *bett* / ___] CMPR]

Once the entries are set up like this, *good* is inserted in the positive (*bett-* does not apply since the conditions for insertion are not met), while *bett-* wins as an exponent of $\sqrt{}$ in the comparative: both *good* and *bett-* are candidates, but *bett-* wins due to Elsewhere (it is more specific). However, the problem is that if there is just a single $\sqrt{}$ in syntax, *bett-* does not win only over *good*, but in fact over any exponent of the $\sqrt{}$ node. This in effect means that in the grammar fragment (48), the comparative of any root is going to be spelled out as *bett-*. Marantz (1995) notes this, and in an attempt to turn this observation into an advantage, he proposed that $\sqrt{}$ suppletion does not exist, and all apparent instances of it involve the exponent of a functional head. If *bett-* is a suppletive exponent of a functional head, then it is not a comparative of the $\sqrt{}$ node, and it does not compete with items like *fast, smart,* etc.

However, Harley (2014) as well as Haugen and Siddiqi (2013) have argued, convincingly to my mind, that $\sqrt{}$ suppletion exists. In order to implement it, and without running into the issues noted above, Harley argues that $\sqrt{}$s must be individuated in syntax (she uses numerical indexes for this purpose, Embick and Noyer 2007 propose that roots carry their phonology and meaning through the entire derivation). However, once $\sqrt{}$s are differentiated in syntax (in whatever way), the initial observation that, e.g., *cat* and *dog* are indistinguishable by the syntactic computation no longer follows. As soon as the syntax contains individuated roots, it

can target them by different sets of rules.[9]

Pointers are important in that they allow us to account for root suppletion without the need to distinguish roots in syntax. To see that, recall that the main reason why numerical indexes on roots were introduced by Harley is to make sure that *bett-* will not win over *fast* (they each spell out a $\sqrt{}$ node with a different index). This is not needed in a cyclic phrasal spell out model with pointers: given the lexical entry for *bett-* in (47-b), there is no reason to expect that this lexical item would outcompete *fast*. In particular, if *fast* had been chosen as the spell out of QP, then (47-b) is simply not a candidate for spell out at all. This is so because the pointer makes sure that *bett-* can only apply when at a previous cycle, *good* has been inserted.

In sum, cyclic spell out with pointers allows one to entertain a theory where individual roots are not distinguished in syntax, and, at the same time, such roots are subject to suppletion, something which is currently impossible in DM.

5 Summary of key features

To sum up, let me repeat here the most important features of the Nano framework as introduced above, starting from the features it shares with DM:

(49) Nanosyntax and DM, shared features:

a. *Late Insertion*: All syntactic nodes systematically lack all phonological features. The phonological features are supplied – after the syntax – by consulting the Vocabulary Items (lexical entries) in the postsyntactic lexicon.

b. *Syntax all the way down*: Terminal nodes are organized into hierarchical structures determined by the principles and operations of the syntax. Since terminal nodes correspond to units smaller than words, it follows that syntactic principles govern the internal structure of words. There is no sharp boundary between the traditional syntax and morphology.

The following properties have been identified as features that differentiate Nano from DM, though not necessarily from other frameworks (like Cartography Cinque and Rizzi 2010 or

[9] I note here that systems with roots individuated in syntax avoid the timing paradox discussed in (10). Since the identity of the root is fixed before the PF—CF split, the PF does not need to directly communicate to CF what choice it had made. Importantly, such approaches are in the minority, so I only mention this in a footnote. The main issue is that these approaches fall short of deriving the 'modularity' observation that roots like *cat* and *dog* are not distinguished in syntax.

work by Kayne 2005, Koopman 2005 and others):

(50) Features differentiating Nano from DM, but not other frameworks:

a. *No Bundling:* The atoms (terminal nodes) of syntactic trees are single features. All combinations of morphosyntactic features arise as the result of (binary) Merge. Pre-syntactic feature bundles do not exist, they correspond to phrases assembled by syntax.

b. *No Morphology.* There is no component of grammar other than syntax that has the power to manipulate syntactic structures. No nodes or features may be added or deleted outside of syntax, no displacement operations take place outside of the syntactic computation.

Finally, the following features are specific to Nano and not present in any other framework, as far as I am aware.

(51) Features specific to Nano:

a. *Cyclic phrasal spellout*. Spell out must successfully apply to the output of every Merge F operation. After successful spellout, the derivation may terminate, or proceed to another round of Merge F, in which case a new round of spellout is initiated, and so on.

b. *Overspecification*. In order for a lexical item to be inserted in a node, the lexical entry must fully contain the syntactic node, including all its daughters, granddaughters, etc., all the way down to every single feature dominated by the node to be spelled out, and in exactly the right geometrical shape.

c. *Spellout-driven movement.* When Merge F produces an FP that cannot be spelled out (no lexical item matches the FP), the FP is rejected at the interface. Syntax then tries to rescue the structure by performing one in a predefined hierarchy of movement operations, before it sends the structure for spellout again. A variety of spellout-driven-movement operations may apply until spellout at FP succeeds. Once lexicalisation succeeds, the derivation either terminates or continues by Merge F.

References

Abels, Klaus and Peter Muriungi. 2008. The focus particle in Kîîtharaka: Syntax and semantics. *Lingua* 118: 687–731.

Anderson, Stephen R. 1992. *A-morphous morphology*, vol. 62. Cambridge University Press, Cambridge.

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

Baker, Mark. 1985. The mirror principle and morphosyntactic explanation. *Linguistic Inquiry* 16: 373–415.

Baunaz, Lena and Eric Lander. 2018a. Deconstructing categories syncretic with the nominal complementizer. *Glossa: a journal of general linguistics* 3 1.

Baunaz, Lena and Eric Lander. 2018b. Nanosyntax: the basics. In *Exploring Nanosyntax*, edited by Lena Baunaz, Karen De Clercq, Liliane Haegeman, and Eric Lander, pp. 3–56. Oxford University Press, Oxford.

Baunaz, Lena and Eric Lander. 2018c. Ontological categories. In *The Unpublished Manuscript*, edited by Pavel Caha, Karen De Clercq, and Guido Vanden Wyngaerd, pp. 1–18. lingbuzz/003993.

Blix, Hagen. 2016. *South Caucasian agreement: A Spanning account*. Master's thesis, University of Vienna.

Bobaljik, Jonathan. 2000. The ins and outs of contextual allomorphy. *University of Maryland Working Papers in Linguistics* 10: 35–71.

Bobaljik, Jonathan. 2012. *Universals In Comparative Morphology*. MIT Press, Cambridge, MA.

Bobaljik, Jonathan and Uli Sauerland. 2018. *ABA and the combinatorics of morphological features. *Glossa* 3: 15.1–34.

Bobaljik, Jonathan David. 2017. Distributed morphology.

Caha, Pavel. 2007. The Superset Principle. Ms., CASTL.

Caha, Pavel. 2009. *The Nanosyntax of Case*. Ph.D. thesis, University of Tromsø, Tromsø.

Caha, Pavel. 2013. Explaining the structure of case paradigms through the mechanisms of Nanosyntax. *Natural Language and Linguistic Theory* 31: 1015–1066.

Caha, Pavel. 2017a. How (not) to derive a *ABA: the case of Blansitt's generalization. *Glossa* 2: 84.1–32.

Caha, Pavel. 2017b. Suppletion and morpheme order: Are words special? *Journal of Linguistics* 53 4: 865–896.

Caha, Pavel. 2018a. Notes on insertion in distributed morphology and nanosyntax. In *Exploring Nanosyntax*, edited by Lena Baunaz, Karen De Clercq, Liliane Haegeman, and Eric Lander, pp. 57–87. Oxford University Press, Oxford.

Caha, Pavel. 2018b. Syncretism as Merge F. Ms., Masaryk University, lingbuzz/004340.

Caha, Pavel, Karen De Clercq, and Guido Vanden Wyngaerd. 2019a. The fine structure of the comparative. *Studia Linguistica* 73 1.

Caha, Pavel, Karen De Clercq, and Guido Vanden Wyngaerd. 2019b. On the difference between $\sqrt{}$ and root. Ms., Lingbuzz.

Caha, Pavel and Marina Pantcheva. 2012. Contiguity beyond linearity. Talk at Decennium: The first 10 years of CASTL.

Calabrese, Andrea. 2008. On absolute and contextual syncretism. In *Inflectional Identity*, edited by Asaf Bachrach and Andrew Nevins, pp. 156–205. Oxford University Press, Oxford.

Chomsky, Noam. 1994. Bare phrase structure. *MIT Occasional Papers in Linguistics* 5: 1–48.

Chomsky, Noam. 2001. Derivation by phase. In *Ken Hale. A Life in Language*, edited by Michael Kenstowicz, pp. 1–52. Cambridge, Massachusetts: MIT Press.

Cinque, Guglielmo. 2005. Deriving Greenberg's universal 20 and its exceptions. *Linguistic Inquiry* 36 3: 315–332.

Cinque, Guglielmo and Luigi Rizzi. 2010. The cartography of syntactic structures. In *The Oxford Handbook of Linguistic Analysis*, edited by Bernd Heine and Heiko Narrog, pp. 51–65. Oxford University Press, Oxford.

Corver, Norbert. 1997. Much-support as a last resort. *Linguistic Inquiry* 28: 119–164.

De Clercq, Karen. 2013. *A unified syntax of negation*. Ph.D. thesis, Ghent University.

De Clercq, Karen and Guido Vanden Wyngaerd. 2017. Splitting up the comparative: evidence from Czech. *Papers of the Linguistic Society of Belgium* 11: 1–18.

De Clercq, Karen and Guido Vanden Wyngaerd. 2018. Unmerging analytical comparatives. *Jezikoslovlje* 19 3.: 341–363.

Dékány, Éva. 2012. *A profile of the Hungarian DP: The interaction of lexicalization, agreement and linearization with the functional sequence*. Ph.D. thesis, Universitetet i Tromsø, Tromsø.

Di Sciullo, Anna-Maria and Edwin Williams. 1987. *On the definition of word*. MIT Press, Cambridge, MA.

Embick, David. 2007. Blocking effects and analytic/synthetic alternations. *Natural Language and Linguistic Theory* 25 1: 1–37.

Embick, David and Rolf Noyer. 2007. Distributed morphology and the syntax/morphology interface. In *The Oxford Handbook of Linguistic Interfaces*, edited by Gillian Ramchand and Charles Reiss, pp. 289–324. Oxford University Press, Oxford.

Halle, Morris. 1997. Distributed morphology: impoverishment and fission. *MIT Working Papers in Linguistics* 30: 425–449.

Halle, Morris and Alec Marantz. 1993. Distributed morphology and the pieces of inflection. In *The View from Building 20*, edited by Ken Hale and Jay Keyser, pp. 111–176. MIT Press, Cambridge, MA.

Halle, Morris and Alec Marantz. 1994. Some key features of distributed morphology. In *Papers on Phonology and Morphology*, edited by Andrew Carnie, Heidi Harley, and Tony Bures, vol. 21 of *MIT Working Papers in Linguistics*, pp. 275 – 288. Cambridge, Mass.

Halle, Morris and Bert Vaux. 1998. Theoretical aspects of Indo-European nominal morphology: The nominal declensions of Latin and Armenian. In *Mir Curad: Studies in Honor of Clavert Watkins*, edited by Jay Jasanoff, H. Craig Melchert, and Lisi Olivier, Innsbrucker Beiträge zur Sprachwissenschaft, pp. 223–240. Institut für Sprachwissenschaft der Universität Innsbruck, Innsbruck.

Hardarson, Gísli Rúnar. 2016. A case for a weak case contiguity hypothesis—a reply to Caha. *Natural Language & Linguistic Theory* 34 4: 1329–1343.

Harley, Heidi. 2014. On the identity of roots. *Theoretical Linguistics* 40: 225–276.

Harley, Heidi and Rolf Noyer. 1999. State-of-the-article: distributed morphology. *Glot International* 4: 3–9.

Harley, Heidi and Elizabeth Ritter. 2002. Structuring the bundle: A universal morphosyntactic feature geometry. In *Pronouns: Grammar and Representation*, edited by Horst J. Simon and Heike Wiese, p. 23–39. John Benjamins, Amsterdam.

Haugen, Jason D. and Daniel Siddiqi. 2013. Roots and the derivation. *Linguistic Inquiry* 44: 493–517.

Haugen, Jason D. and Daniel Siddiqi. 2016. Towards a restricted realization theory: Multimorphemic monolistemicity, portmanteaux, and post-linearization spanning. In *Morphological metatheory*, edited by Daniel Siddiqi and Heidi Harley, pp. 343–385. John Benjamins, Amsterdam.

Holaj, Richard. 2018. Balancing between roots and thematic vowles. In *The unpublished manuscript*, edited by Pavel Caha, Karen De Clercq, and Guido Vanden Wyngaerd, pp. 81–93. Lingbuzz, lingbuzz/003993.

Jakobson, Roman. 1936 [1962]. Beitrag zur allgemeinen Kasuslehre: Gesamtbedeutungen der russischen Kasus. In *Selected writings, vol. 2*, pp. 23–71. Mouton, The Hague.

Johnston, Jason. 1996. *Systematic Homonymy and the Structure of Morphological Categories. Some Lessons from Paradigm Geometry*. Ph.D. thesis, University of Sydney.

Kayne, Richard. 1994. *The Antisymmetry of Syntax*. MIT Press, Cambridge, MA.

Kayne, Richard. 2005. Some notes on comparative syntax: With special reference to English and French. In *Movement and Silence*, pp. 277–333. Oxford University Press, Oxford.

Kayne, Richard S. 2010. Toward a syntactic reinterpretation of Harris & Halle (2005). In *Selected papers from "Going Romance," Groningen 2008*, edited by R. Bok-Bennema, B. Kampers-Manhe, and B. Hollebrandese, pp. 145–170. John Benjamins, Amsterdam.

Kiparsky, Paul. 1973. 'Elsewhere' in phonology. In *A Festschrift for Morris Halle*, edited by Stephen Anderson and Paul Kiparsky, pp. 93–106. Holt, Rinehart & Winston, New York.

Koopman, Hilda. 2005. Korean (and Japanese) morphology from a syntactic perspective. *Linguistic Inquiry* 36 4: 601–633.

Křivan, Jan. 2012. Komparativ v korpusu: explanace morfematické struktury českého stupňování na základě frekvence tvarů. *Slovo a slovesnost* 73: 13–45.

Lander, Eric and Liliane Haegeman. 2018. The nanosyntax of spatial deixis. *Studia linguistica* 72 2: 362–427.

Marantz, Alec. 1991. Case and licensing. In *Proceedings of ESCOL '91*, edited by Benjamin Ao German Westphal and Hee-Rahk Chae, pp. 234–253. Cornell Linguistics Club, Ithaca.

Marantz, Alec. 1995. A late note on late insertion. In *Explorations in generative grammar: A festschrift for Dong-Whee Yang*, edited by Young Sun Kim, Byung-Choon Lee, Kyoung-Jae Lee, Kyun-Kwon Yang, and Jong-Kuri Yoon, pp. 396–413. Hankuk, Seoul.

Marantz, Alec. 1997. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. In *University of Pennsylvania Working Papers in Linguistics*, edited by Alexis Dimitriadis, Laura Siegel, Clarissa Surek-Clark, and Alexander Williams, vol. 4, pp. 201–225. University of Pennsylvania.

Matushansky, Ora. 2013. More or better: On the derivation of synthetic comparatives and superlatives in English. In *Distributed Morphology Today: Morphemes for Morris Halle*, edited by Ora Matushansky and Alec Marantz, pp. 59–78. MIT Press, Cambridge, MA.

McCawley, James. 1968. Lexical insertion in a transformational grammar without deep structure. *CLS* 4: 71–80.

McCreight, Katherine and Catherine V. Chvany. 1991. Geometric representation of paradigms in a modular theory of grammar. In *Paradigms: The Economy of Inflection*, edited by Frans Plank, pp. 91 – 112. Mouton de Gruyter, Berlin, New York.

McFadden, Thomas. 2004. *The position of morphological case in the derivation*. Ph.D. thesis, UPenn.

McFadden, Thomas. 2017. *ABA in stem-allomorphy and the emptiness of the nominative. *Glossa* .

Merchant, Jason. 2015. How much context is enough?: Two cases of span-conditioned stem allomorphy. *Linguistic Inquiry* 46: 273–303.

Neeleman, Ad and Kriszta Szendrői. 2007. Radical pro-drop and the morphology of pronouns. *Linguistic Inquiry* 38: 671–714.

Newell, Heather and Maire Noonan. 2018. A re-portage on spanning; feature portaging and non-terminal spell-out. In *McGill Working Papers in Linguistics*, pp. 156–167. McGill University, Montreal.

Pantcheva, Marina. 2010. The syntactic structure of locations, goals, and sources. *Linguitics* 48: 1043–1081.

Plank, Frans. 1991. Rasmus Rask's dilemma. In *Paradigms: The Economy of Inflection*, edited by Frans Plank, pp. 161–196. Mouton de Gruyter, Berlin.

Radkevich, Nina. 2010. *On Location: The structure of case and adpositions*. dissertation, University of Connecticut, Storrs, CT.

Ramchand, Gillian. 2008. *Verb Meaning and the Lexicon*. Cambridge University Press, Cambridge.

Siddiqi, Daniel. 2006. *Minimize Exponence: Economy Effects on the Morphosyntactic Component of the Grammar*. Ph.D. thesis, University of Arizona.

Smith, Peter W., Beata Moskal, Ting Xu, Jungmin Kang, and Jonathan David Bobaljik. 2018. Case and number suppletion in pronouns. *Natural Language and Linguistic Theory* .

Starke, Michal. 2002. The day syntax ate morphology. Class taught at the EGG summerschool, Novi Sad.

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

Starke, Michal. 2014a. Cleaning up the lexicon. *Linguistic Analysis* 39: 245–256.

Starke, Michal. 2014b. Towards an elegant solution to language variation: Variation reduces to the size of lexically stored trees. In *Linguistic Variation in the Minimalist Framework*, edited by Carme M. Picalo, pp. 140 – 153. Oxford University Press, Oxford.

Starke, Michal. 2017. Resolving (dat = acc) $\neq$ gen. *Glossa* 2 1: 104.1–8.

Starke, Michal. 2018. Complex left branches, spellout, and prefixes. In *Exploring Nanosyntax*, edited by Lena Baunaz, Karen De Clercq, Liliane Haegeman, and Eric Lander, pp. 239–249. Oxford University Press, Oxford.

Svenonius, Peter. 2012. Spanning. Ms., CASTL.

Taraldsen, Knut Tarald, Lucie Taraldsen Medová, and David Langa. 2018. Class prefixes as specifiers in southern bantu. *Natural Language & Linguistic Theory* 36 4: 1339–1394.

Taraldsen, Tarald. 2012. *ABA and the representation of features in syntax. Talk presented at BCGL 7, Brussels.

Taraldsen Medová, Lucie and Bartosz Wiland. 2018. Semelfactives are bigger than degree achievements. *Natural Language & Linguistic Theory* .

Van Baal, Yvonne and Jan Don. 2018. Universals in possessive morphology. *Glossa: a journal of general linguistics* 3 1.

Vanden Wyngaerd, Guido. 2018. The feature structure of pronouns: a probe into multidimensional paradigms. In *Exploring Nanosyntax*, edited by Lena Baunaz, Karen De Clercq, Liliane Haegeman, and Eric Lander, pp. 277–304. Oxford University Press, Oxford.

Weerman, Fred and Jacqueline Evers-Vermeul. 2002. Pronouns and case. *Lingua* 112: 301–338.

Zompì, Stanislao. 2017. *Case decomposition meets dependent-case theories*. Master's thesis, Pisa: Universita de Pisa.