Word-order information in the lexicon

Francesco Pinzin (University of Padua) & Tommaso Mattiuzzi (Goethe University Frankfurt)

Crosslinguistic variation in basic word-order as (1) necessarily depends on differences in the word-order information stored in the speakers' competence. What is crucial is where such information is stored, in what format, and how it interacts with the rest of the grammar to give rise to the attested typological patterns (e.g., U20 generalisations, cf. Abels, 2016; Cinque, 2005; Greenberg, 1963).

(1) a. *black dog*

b. *cane nero* (Italian)
dog black

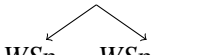
We propose that word-order information is stored in the lexicon, in the format of Lexical Items (LIs), and that the operations yielding word-order are the same that guide lexicalisation more generally in Nanosyntax, encoded in the Lexicalisation Algorithm (LA, Starke, 2009 and ff.). The LA replicates Cinque's movements and therefore the main result in the existing literature, namely the restriction of derivational possibilities to all and only the attested patterns (Abels, 2016; Cinque, 2005, 2009). Unlike current analyses (Abels and Neeleman, 2012; Cinque, 2023), this lexicalisation-based approach makes no reference to (traditional) syntactic movement, sparing the system from stipulating semantically vacuous structural dependencies triggering movement.

In our proposal, the word-order properties of a language result from the interaction between the universal functional sequence (*fseq*), the universal LA in (2) (a simplified version, see Caha et al., 2023 for details) and the set of available LIs. As per basic Nanosyntactic assumptions, each LI is a stored link between phonological and conceptual information and a syntactic tree defining its lexicalisation properties. No intrinsic principle dictates the simultaneous presence of all three types of information in each LI, which predicts the possibility of LIs that only contain syntactic information. We argue that this type of LIs is what stores language-specific word-order information, and that a string like (1a) requires a lexicon containing the LI in (3).

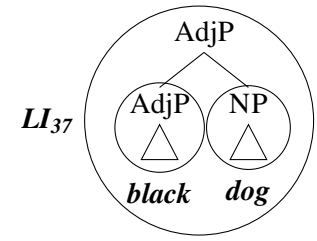
(2) Lexicalisation Algorithm:

1. MERGE *f*	a. LEXICALISE [FP].
2. MERGE FP	b. LEX-DRIVEN MOV I: If fail, evacuate the closest labelled non-remnant constituent, re-try a.
	c. LEX-DRIVEN MOV II: If fail, evacuate the immediately dominating constituent, re-try a. (recursive)

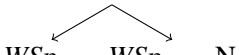
(3) ⟨ LI_{37}: AdjP ⟩



(4)



(5) LI_{78}: ⟨ DemP ⟩



To see how, consider the relation between structure-building and lexicalisation operations formalised in (2). At each cycle, the derivation merges into the current workspace (*WSp*) a feature *f* (MERGE *f*). The resulting syntactic structure (S-tree) must be licensed via lexicalisation (steps a-c.). In our implementation, lexicalisation is successful when all labelled (sub)trees contained in the S-tree are matched by an LI. A tree is matched iff the lexicon contains an LI that stores an identical tree (L-tree) (Caha et al., 2023). If all attempts at lexicalisation fail (steps a-c.), the next option is to try to add *f* in a different way, that is by merging a tree containing *f* derived in a

parallel workspace (MERGE FP). Like MERGE *f*, MERGE FP derives a new labelled tree, which must therefore be lexicalised, triggering steps a-c.

LIs like (3) encode word-order information by storing a syntactic arrangement of the two XPs involved in a MERGE FP operation: the main one requiring *f* and the XP providing *f* (FP). They do so by pointing to the independent *WSp*s in which each XP is built (WSp_{main} & WSp_{FP}), as signalled by the arrows. Pointers can be conceived as breakpoints in the matching process: once the comparison between an S-tree and an L-tree finds a pointer to some node *N* in the L-tree, a secondary comparison process evaluates whether the corresponding node in the S-tree is matched based on the information stored in *N*. If $N = WSp_{\text{x}}$, matching is evaluated based on the last lexicalised S-tree in WSp_{x}. Being void of phonological and conceptual content, LIs like (3) inherit the phonological and conceptual content of their branches.

Concretely, (3) matches any AdjP whose left branch contains the last lexicalised S-tree in WSp_{Adj}, and whose right branch contains the last lexicalised S-tree in WSp_{main}. By this, (3) matches structures of any degree of complexity, provided that the label of the root node is AdjP and the two *WSp*s are in the configuration defined by the LI. Hence, if the lexicon of a language contains (3), the order [Adj>N] is licensed (the case of (1a), shown in (4)). If no such LI is present, step a. fails and the following steps of the LA are triggered, which dictate lexicalisation-driven movements of increasingly larger constituents. Two further scenarios are given. The first is (sub)extraction, which strands a part of the complement of the newly merged branch (yielding orders like N-Dem-Num-Adj). Configurations derived by (sub)extraction are matched by LIs like (5), containing diacritics like *-N*. These specify a category in the *fseq* that identifies a portion of the tree in WSp_{main} that is ignored for the sake of matching. This portion corresponds to any node dominated by the specified category or a lower one in the *fseq*. If what remains in WSp_{main} matches the right branch of the S-tree (which can only happen after (sub)extraction applies), lexicalisation is successful. The second scenario is 'roll-up', and arises when the entire complement of the new branch is moved because no LI matches previous movement steps (the case of (1b)). As we will show, 'roll-up' configurations are lexicalised without matching additional LIs, which entails that full 'roll-up' always leads to a converging derivation, (i.e., full 'roll-up' is a last-resort option).

Under this analysis, the U20 generalisation results from the limited nature of the movement options defined by the LA, in line with Cinque's approach. Each possible order amounts to a different balance among two opposite tendencies: minimise movements vs. minimise the number of LIs. The first entails more LIs to license configurations without triggering movement, the latter requires more movements without the need for additional licensing LIs. This lexicalisation-based system avoids the stipulation of semantically vacuous syntactic dependencies, and provides a rationale for why these operations obey different constraints as traditional syntactic movement. Finally, it allows to maintain a universal linearisation procedure (in line with Cinque, 2023) and a single locus for syntactic variation, i.e., the lexicon.

Abels, Klaus (2016), "The Fundamental LeftRight Asymmetry in the Germanic Verb Cluster".• Abels, Klaus and Ad Neeleman (2012), "Linear Asymmetries and the LCA: Linear Asymmetries and the LCA".• Caha, Pavel et al. (2023), *Nanosyntax: State of the Art and Recent Developments*, pre-published.• Cinque, Guglielmo (2005), "Deriving Greenberg's Universal 20 and Its Exceptions".• — (2009), "The Fundamental Left-Right Asymmetry of Natural Languages".• — (2023), *On Linearization: Toward a Restrictive Theory*.• *Universals of Language* (1963).• Starke, Michal (2009), "Nanosyntax: A Short Primer to a New Approach to Language, Ms."•