Strict Locality in Syntax

Keywords: computational complexity, strictly local languages, selection, functional hierarchy, adjunct order

1. Overview One major goal of linguistic theory is to account for restrictions on the locality of linguistic dependencies: over what distance may they hold, and under what conditions? Computational complexity provides a natural and insightful way to describe such restrictions. Recent work has accumulated substantial evidence that all linguistic patterns fall within very simple classes of formal languages, known as *subregular* languages. These are string languages in the case of phonology and morphology, but tree languages for syntax (Graf and Heinz 2015; Graf 2018). Among the subregular classes implicated for natural language is the class of *strictly local* (SL) languages, which have been used to model local phonotactics (e.g. syllable structure) as well as category selection in syntax.

However, the pervasiveness of SL in syntax has not been fully appreciated: in addition to selection, functional hierarchies are SL, as is adjunct ordering, and perhaps also certain kinds of last resort operations. All of these can be understood as instantiations of a single structure building operation. Within syntax, this operation plausibly corresponds to Chomsky's (1995) Merge, but the present perspective suggests a broader conclusion: SL computations are the basis for all linguistic structure. This in turn has significant implications for language cognition.

2. Introduction to SL An SL-k string language is characterized by the following property: a string is ruled out if it contains any banned substrings of length k, equivalently it is ruled in if every such substring is licit. For illustration, consider a (simplified) model for Japanese phonotactics: the syllable template is (C)–(j)–V–(N), where 'C' stands for a consonant, 'V' for a vowel, 'j' for a palatal glide, and 'N' for a nasal coda. This pattern is SL-2. A positive grammar, which lists all licit substrings, is given on the left side of Figure 1 below. We can visualize this grammar using its finite state automaton (FSA) representation, shown on the right. Every path through the FSA which ends in a final state (marked with double circles) represents a string in the language. While the full class of languages implemented by FSAs (the regular languages) is much larger than SL, all shown here are within SL. For an SL-2 grammar, each state is named with a single symbol, and its outgoing transitions with the symbols that may follow.

Grammar: {**$C**, $j, $V, **Cj**, **CV**, **jV**, VC, Vj, **VN**, V$, **NC**, Nj, NV, **N$**}

Example word: sjunkan 'moment'

Substrings: $s sj ju un nk ka an n$

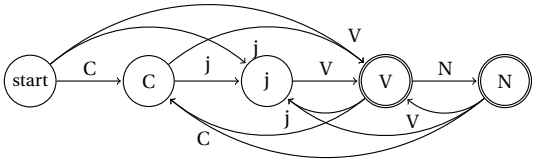


Figure 1: Japanese phonotactics is SL-2. Left: positive SL-2 grammar and example word, with attested substrings highlighted. $ marks the beginning/end of a word. Right: FSA representation of grammar.

3. SL in syntax, selection To show that selection, functional hierarchies, and adjunct ordering are all SL, I make use of Minimalist Grammar (MG) dependency trees. MGs (Stabler 1997) are a formalization of ideas from Chomsky's (1995) Minimalist Program. Following recent precedent (cf. Graf 2022; Graf and Shafiei 2019), I utilize dependency trees since they are compact while containing all necessary information about the derivation. As an example, the dependency tree for the sentence "The cat chased the rat" is shown in Figure 2. The rightmost child of each node is its complement; others are specifiers. Positive features are "selector" features and negative features are "category" features. For example, the feature specification for the v head is $\langle \text{lv}^- \ \text{D}^+ \ \text{V}^+ \rangle$, encoding the fact that v takes a DP specifier and a VP complement. I omit all movement features since they are not relevant here.

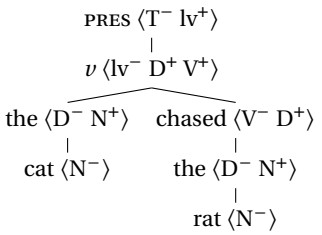
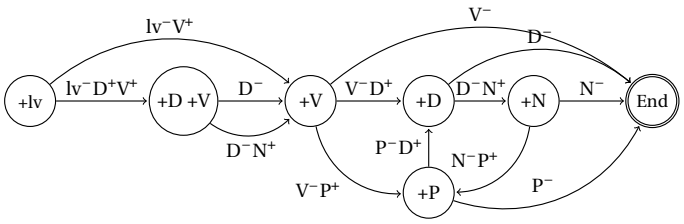


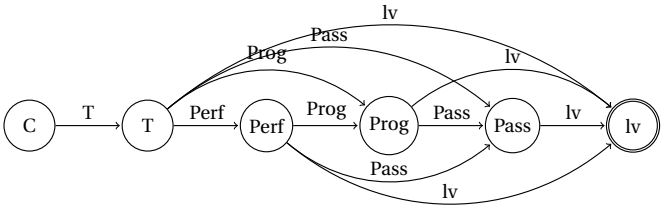
Figure 2: MG dependency tree for "The cat chased the rat."

Now, consider the path from the root to the rightmost leaf which includes all nodes which dominate or c-command the leaf—call this the *spine* of the tree (cf. Graf and De Santo 2019). The spine for our example is PRES · *v* · the · chased · the · rat. This string forms the basis for the SL computation of syntactic dependencies (note that each left branch begins a new spine). For example, the FSA in Figure 3a implements an SL-3 string language which handles selection for a fragment of English.

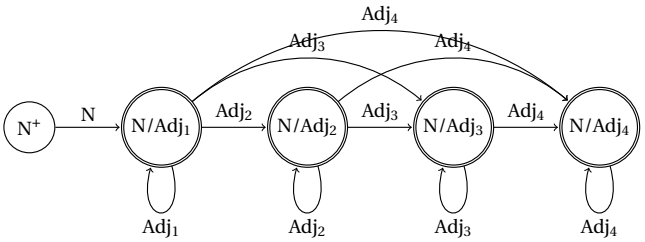
4. Functional hierarchies Functional hierarchies are theoretically problematic due to their properties of strict ordering and optionality. If modeled as feature-driven selection, each head must allow a variety of selectional frames which conspire to enforce the correct order. But mathematically, a functional hierarchy is just a *preorder* (cf. Larson 2021), which is in turn an SL-2 pattern: having seen one node, we know which nodes may follow. In this case it is natural to label the states and transitions with just the category of the node. Such an FSA for the English clausal hierarchy is shown in Figure 3b. We could also label the transitions with MG feature lists, recapitulating the "multiple selectional frames" analysis, but the shape of the machine is identical. Furthermore, the structure of this pattern—a series of slots in fixed order, some of which are optionally empty—is parallel to syllable structure.



(a) FSA for lexical selection in English.



(b) FSA for English clausal hierarchy.



(c) FSA schematic for adjunct ordering.

Figure 3: FSAs for three SL syntactic patterns.

5. Adjunct ordering Adjunct ordering, e.g. the fact that *big blue house* is more natural than *blue big house*, has a similar structure to a functional hierarchy. The difference is that the FSA also contains loops, allowing structures like *big big big blue house*. For concreteness, I treat adjuncts as dependents of their head. A schematic FSA is shown in Figure 3c. The same structure arises if adjuncts are instead analyzed as being introduced by functional heads, as in cartographic approaches (cf. Cinque 1999). Thus, the claim that adjunct ordering is SL is robust against variation in analytical choices.

6. Implications Because SL is among the very weakest subregular classes, this means that the most fundamental aspects of linguistic structure building are also among the simplest and most local. Furthermore, non-local dependencies such as consonant harmony (Heinz 2018) and syntactic movement (Graf 2022) are also subregular. Within syntax at least, these appear to be *tier-based strictly local*, or TSL, a generalization of SL in which certain elements are ignored. Thus, both local and non-local dependencies arise from very similar, simple computations. These findings help to explain why such patterns exist, and also suggest how such patterns may be learned (cf. Lambert et al. 2021).

References **Cinque, Guglielmo** (1999). Adverbs and functional heads: A cross-linguistic perspective. Oxford University Press. · **Chomsky, Noam** (1995). The Minimalist Program. MIT Press. · **Graf, Thomas** (2022). "Typological Implications of Tier-Based Strictly Local Movement". SCiL 2022. · **Graf, Thomas and Aniello De Santo** (2019). "Sensing Tree Automata as a Model of Syntactic Dependencies". MOL 16. · **Graf, Thomas and Jeffrey Heinz** (2015). Commonality in Disparity: The Computational View of Syntax and Phonology. GLOW 2015. · **Heinz, Jeffrey** (2018). "The computational nature of phonological generalizations". Phonological Typology, Phonetics and Phonology. · **Larson, Richard K.** (2021). "Rethinking cartography." Language 97. · **Stabler, Edward P.** (1997). "Derivational minimalism". Logical Aspects of Computational Linguistics. Springer.