Proceedings of the Society for Computation in Linguistics

Volume 3 Article 35

2020

Curbing Feature Coding: Strictly Local Feature Assignment

Thomas Graf
Stony Brook University, mail@thomasgraf.net

Curbing Feature Coding: Strictly Local Feature Assignment

Thomas Graf
Department of Linguistics
Stony Brook University
Stony Brook, NY 11794, USA
mail@thomasgraf.net

Abstract

Graf (2017) warns that every syntactic formalism faces a severe overgeneration problem because of the hidden power of subcategorization. Any constraint definable in monadic second-order logic can be compiled into the category system so that it is indirectly enforced as part of subcategorization. Not only does this kind of feature coding deprive syntactic proposals of their empirical bite, it also undermines computational efforts to limit syntactic formalisms via subregular complexity. This paper presents a subregular solution to feature coding. Instead of features being a cheap resource that comes for free, features must be assigned by a transduction. In particular, category features must be assigned by an input strictly local (ISL) tree-to-tree transduction, defined here for the first time. The restriction to ISL transductions correctly rules out various deviant category systems.

1 Introduction

Theoretical and computational linguists both strive to identify limited models of language that furnish sufficient power without allowing for excessive overgeneration. Recently, Graf (2017) noted that the findings of Graf (2011) and Kobele (2011) point towards a major loop hole in all current theories of syntax. The category system can be abused to encode additional information about the syntactic tree, and the usual subcategorization requirements can then be used to enforce a certain kind of synchronization between parts of the tree. For instance, the category DP may be split into DP[+NPI] and DP[−NPI] depending on whether the DP is an NPI, and the category X of each selecting head becomes X[+NPI] if the argument it selects contains an unlicensed NPI. This simple strategy has been known for a long time but did not raise serious concerns as it is widely accepted that all grammar formalisms "leak" in the sense that they also allow for some unnatural patterns.

But the extent of the problem for linguistic theory has not been fully appreciated. Graf (2017) shows how this strategy can be generalized to flout all island constraints, enforce constraints that lack any notion of locality, and even add highly unnatural counting requirements to the grammar. Every constraint that can be defined in monadic second-order logic is expressible through category refinement. This allows for very unnatural constraints, e.g. enforcing verb-second word order iff the sentence contains exactly three relative clauses or both a Principle A violation and a word in which unbounded tone plateauing is not obeyed. The only way to preclude this is to restrict the shape of category systems, but Graf (2017) argues that the usual linguistic requirements on syntactic categories are insufficient. Hence every syntactic formalism lacks a key mechanism to distinguish natural patterns from unnatural ones, resulting in massive overgeneration.

This paper proposes a computational solution to this problem, drawing from recent work on subregular complexity. Features no longer come part and parcel with lexical items, but must be assigned to tree structures by a transduction. An unnatural feature system that keeps track of, say, a counting dependency, requires a very powerful transduction. The category systems of natural languages, on the other hand, can be handled by much simpler means. I argue that these category systems only require inspection of a lexical item's local context. This intuition is formalized by generalizing the input-strictly local (ISL) string-to-string mappings of Chandlee (2014) to ISL tree-to-tree transductions. To the best of my knowledge, this is the first time a subregular tranduction class is defined for trees, and I hope it will be a fertile vantage point for mathematical and empirical work alike.

Proceedings of the Society for Computation in Linguistics (SCiL) 2020, pages 362-371.
New Orleans, Louisiana, January 2-5, 2020

The paper deviates slightly from the usual structure. Since the problem is also of interest to theoretical linguists and the proposed solution is fairly intuitive, the first half focuses on the big picture and keeps formal concepts to a minimum (§2). The mathematical aspects are then worked out in §3, the most important of which is the formal definition of ISL tree transductions (§3.2).

2 Problem and Solution: Informal Sketch

The power of category systems and subcategorization is best illustrated with an example (§2.1). This makes it clear what unnatural category systems may look like, and in what respects they clearly differ from natural ones (§2.2). The problem of category abuse in syntax is actually an instance of the more general phenomenon of feature coding, which also appears in the domain of subregular complexity (§2.3). But subregular complexity also provides a way of measuring the complexity of feature systems via transductions. With strict limits on the power of these transductions, many of the unnatural category systems are correctly ruled out (§2.4) while it becomes possible to formulate new syntactic universals (§2.5).

2.1 A Grammar with Odd/Even Counting

Let us start with a toy example from Minimalist grammars (MGs; Stabler, 1997, 2011) that illustrates the power of syntactic categories. MGs are closely modeled after Minimalist syntax, and subcategorization is encoded via category and selector features that drive the operation *Merge*. A head with *selector feature* X^+ can only be merged with a phrase whose head has *category feature* X^-. This matching of features is called *feature checking*. The category feature of a lexical item l can only be checked once all selector features of l have been checked. While exceedingly simple, this system is already too powerful as a model of subcategorization in natural languages.

Consider the MG G where the only pronounced lexical items are *foo* and *bar*, which may have the category features E^- or O^-. By default, the category feature is O^-. But if the lexical item carries a selector feature O^+ or E^+, the category feature must be the opposite of that selector feature (E^- or O^-, respectively). Hence *foo* and *bar* may have the feature strings O^-, $\mathrm{E}^+\mathrm{O}^-$, or $\mathrm{O}^+\mathrm{E}^-$. Besides *foo* and *bar*, the MG only has an unpronounced C-head, which must always be the last lexical item to be merged. The C-head carries the selector feature O^+. Overall, G consists of the following lexical items:

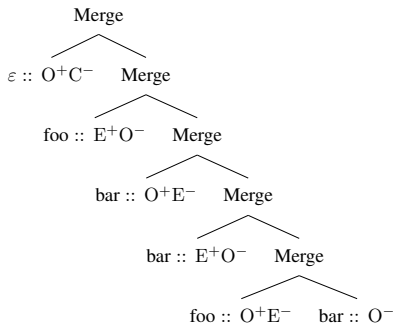

Figure 1: Derivation tree for *foo bar bar foo bar*

(1) **MG G with even/odd alternation**

$$\begin{array}{lll} \varepsilon :: \mathrm{O}^+\mathrm{C}^- & \text{foo} :: \mathrm{E}^+\mathrm{O}^- & \text{foo} :: \mathrm{O}^- \\ & \text{foo} :: \mathrm{O}^+\mathrm{E}^- & \\ & \text{bar} :: \mathrm{E}^+\mathrm{O}^- & \text{bar} :: \mathrm{O}^- \\ & \text{bar} :: \mathrm{O}^+\mathrm{E}^- & \end{array}$$

The MG generates any string over *foo* and *bar* whose length is odd. The reasoning for this is as follows: the derivation must start with either foo :: O^- or bar :: O^-. From this point on, selecting heads alternate between E^- and O^-, but only a head carrying O^- can be selected by the C-head to end the derivation. The end result is that the number of pronounced lexical items in the tree must be odd, as is also illustrated in Fig. 1. The MG above thus instantiates a simple case of *modulo* counting at the string level.

2.2 (Un)Naturalness of the Example MG

The example grammar G in (1) is highly unnatural in several respects. First of all, string length does not seem to be a relevant criterion for natural language syntax. This definitely holds for *modulo* counting, which is unheard of. But even absolute size requirements are hard to come by unless one abandons the well-motivated competence-performance distinctions. A potential counterexample is Heavy NP-shift, which is sensitive to a constituent's size and thus, possibly, its string length. But even here processing provides a more plausible explanation (cf. Liu, 2018). Syntax itself seems to be completely blind to size, be it string length or the size of a tree.

Perhaps even more important is the fact that O^- and E^- do not convey intrinsic information of the

lexical item l that carries them. Instead, these categories represent properties of the whole subtree. Hence the category is highly context-dependent. If one wanted to insert another instance of *foo* or *bar* in the subtree headed by l, one would also have to change the category of l because of how O^- and E^- have to alternate. The change of l's category then requires changing the category of l's selector, the selector of l's selector, and so on. This directly contradicts a basic principle of selection: a lexical item selects for its argument, not the argument(s) of its argument. A verb selecting a PP may restrict the shape of the P-head, but not the DP inside the PP. And no lexical item can freely select any head of any category as long as the selected subtree satisfies some other property. Subcategorization enforces head-head dependencies, not head-subtree dependencies, and any category system that allows the latter to be reduced to the former is missing a key aspect of natural language.

2.3 The Full Extent of the Problem

As was already mentioned in the introduction, the example above is but the tip of the iceberg. Without restrictions on the category system, any arbitrary constraint can be enforced as long as it is definable in monadic second-order logic. Graf (2017, p. 22–24, p. 27f) gives several illustrative examples of overgeneration and explains in detail why the usual heuristics (e.g. syntactic distribution, morphological inflection) are not sufficient to distinguish natural from unnatural category systems. Beyond *modulo* counting, this kind of *feature coding* also allows for, among other things, strange constraint interactions ("Satisfy either verb-second or Principle A, but not both"), symmetric counterparts of existing constraints (Reverse Principle A: every reflexive must c-command a suitable R-expression), and displacement mechanisms that do not use movement and hence bypass island constraints. All of this becomes possible because feature coding abuses categories as a local buffer for non-local information, erasing all locality and complexity differences between constraints.

The potential abuse of syntactic categories is actually an instance of a more general problem that has to be carefully avoided in subregular phonology. Subregular phonology (see Heinz 2018 and references therein) has identified very restricted subclasses of the regular string languages that still furnish enough power for phonology. Crucially, though, these claims depend on the choice of features because every regular pattern can be made subregular by introducing additional features. In formal terms: every recognizable set is a projection of a local set (cf. Rogers, 1997).

For instance, the regular string language of odd-length strings over a can be pushed into the extremely weak subclass of *strictly 2-local* string languages if one introduces a feature $[\pm\text{odd}]$. A string like $a\ a\ a\ a\ a$ would then be represented as $a[+\text{odd}]\ a[-\text{odd}]\ a[+\text{odd}]\ a[-\text{odd}]\ a[+\text{odd}]$. The language with the diacritic $[\pm\text{odd}]$ feature is strictly 2-local because it can be expressed in terms of constraints that involve at most two segments:

(2) **Strictly 2-local constraints**

a. Every string must start with $a[+\text{odd}]$ and end with $a[+\text{odd}]$.

b. $a[+\text{odd}]$ must not follow $a[+\text{odd}]$.

c. $a[-\text{odd}]$ must not follow $a[-\text{odd}]$.

The example in §2.1 is a syntactic analog of this trick, with O^- and E^- filling the roles of $[\pm\text{odd}]$. In all these cases, feature coding obfuscates subregular complexity by precompiling complex dependencies into an invisible alphabet of features and diacritics.

The feature coding problem is less severe in subregular phonology thanks to the restriction to articulatory features, which can usually be replaced by the actual segments without changing anything substantive about the analysis.[1] In syntax, features play a much more vital role as two representations may look exactly the same except for their feature make-up.

For instance, Fig. 2 gives an MG dependency tree representation for *the gardeners water their flowers*, while adding the movement features top^- to *their* and top^+ to *water* yields the MG dependency tree representation of the very different topicalization sentence *their flowers, the gardeners water*. The movement features are an essential part of the representation. Similarly, category and selector features can be crucial for head-argument relations in MG derivation trees. In Fig. 1, switching the feature strings of the bottom-most *foo* and

[1] One notable exception is Baek (2018). She adds a limited number of structural features to define a subregular class that lies strictly between the classes TSL (Heinz et al., 2011) and ITSL (De Santo and Graf, 2019).

bar would yield a new string *foo bar bar bar foo*. This is because derivation trees encode head-argument relations only via Merge features, not via dominance or linear order. It is not surprising, then, that all the recent work extending the subregular perspective from phonology to syntax relies on feature in one way or another (Graf, 2018; Graf and Shafiei, 2019; Graf and De Santo, 2019; Vu, 2018; Vu et al., 2019).

But even if features could be done away with, that would be too extreme a step as they can still be useful. Consider once more the case of topicalization movement. This involves three computational steps: I) identifying the mover and the target site, II) determining whether topicalization movement is licit, and III) displacing the topicalized phrase. Without features, the first two steps would have to be handled by the same computational device, which first makes a non-deterministic choice as to what should move where, and then decides whether this instance of movement obeys all relevant constraints. By making features an integral part of the representation, we factor out the first step in order to isolate the complexity of the second step. But without a restrictive theory of features, there is the risk of factoring out more than intended. This would lead to misleading claims about subregular complexity that are merely artifacts of feature coding. Subregular syntax thus finds itself in a precarious situation where the very thing it depends on also threatens to undermine all its findings.

The original problem of syntactic categories thus is but a piece of the larger puzzle of how to avoid feature coding. The brute force solution of shunning features altogether is not workable in syntax. Features distinguish otherwise identical representations; theoretical and computational linguists alike are too accustomed to thinking in terms of features; and features do allow for insightful factorizations of complexity. The problem is not features as such, it is the lack of a measuring rod for how much complexity has been shifted into the feature system.

2.4 Solution: Strictly Local Feature Assignment

Features come for free under current models of complexity because they are representational devices. Subregular complexity takes the representations for granted and then investigates how hard a given dependency would be to enforce over these representations. In order to assess the complexity of feature systems, we have to decouple them from the representations. Intuitively speaking, we want to measure the complexity of constructing a feature-annotated representation from its feature-free counterpart.

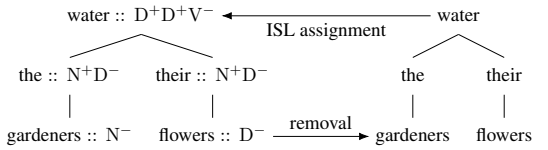

Figure 2: Feature assignment as a transduction problem between a feature-annotated MG dependency tree (left) and its feature-free counterpart (right)

Formally, this takes the shape of a transduction problem. For strings, transductions are a formal counterpart of rewrite rules, and for trees they are similar to syntactic transformations in the sense of Chomsky (1965). Chandlee (2014) defines a particularly weak kind of string transductions known as *input strictly local* (ISL). An ISL transduction considers only the local context of a symbol when deciding how it should be rewritten. Word-final devoicing and intervocalic voicing are examples of ISL transductions in phonology, whereas long-distance sibilant harmony would not be ISL because the rewriting of a sibilant can depend on other segments that are arbitrarily far away. ISL can be lifted from strings to trees: a node in a tree may be rewritten in various ways depending on its local context in the tree. A transduction is ISL-k iff all local contexts can be limited to at most k levels (a mother-daughter configuration, for example, involves two levels).

Figure 2 illustrates the approach with a feature-annotated MG dependency tree for *the gardeners water their flowers*. The question at hand is whether the familiar categories D, N, and V can be assigned by an ISL transduction. We take a feature-annotated representation like the one of the left and remove all category and selector features. Then we have to define an ISL transduction that takes us back to the original representation. If this can be done with any well-formed tree, then the whole feature system is *ISL recoverable*.

For the specific tree in Fig. 2, we need an ISL-2 transduction. The feature annotations for *the*, *their*, and *gardeners* can be recovered without any further context information just from the phonetic

exponents. That is the case because there simply are no alternative feature annotations for these lexical items in English. With *water* and *flowers*, on the other hand, there is ambiguity as each one of them could be either a noun or a verb. But in both cases a minimum amount of context is sufficient to disambiguate their categories. Since *flowers* is selected by *the*, which can only be a determiner, *flowers* must be a noun. Similarly, *water* must be a verb because it selects *the* and *their*, neither one of which could be an argument of the noun *water*. Inspecting the daughters or the mother of a node requires a context with two levels, so the transduction is ISL-2 for this specific example. The complexity of the whole feature system corresponds to the weakest transduction that works for all well-formed trees (usually there will be infinitely many of those; therefore, conclusive complexity results require proofs rather than examples).

The feature system of the MG G in Sec. 2.1 is not ISL recoverable. This follows from the fact that it is not ISL-k recoverable for any $k \geq 1$. For the sake of simplicity, we will once again use a dependency tree format as in Fig. 2 instead of the derivation tree format in Fig. 1. Now suppose that the features for the left tree in Fig. 3 could be correctly assigned from the middle tree by an ISL-k transduction. Since the transduction is ISL-k, the features assigned to *foo* depend exclusively on some context with at most k levels. Crucially, *foo* will always receive the same features as long as the context remains the same. But now compare this to the tree on the right. Here *foo* has switched positions with *bar* below it, inducing a change in its feature make-up. Yet the locally bounded context for *foo* has not changed at all — the middle tree could also be a description for the right tree depending on the values of $m' \geq k$ and $n' \geq k$. Hence the feature annotation for *foo* varies despite identical contexts, which proves that the feature system is not ISL recoverable. In fact, no ISL transduction can handle any feature system that involves *modulo* counting.

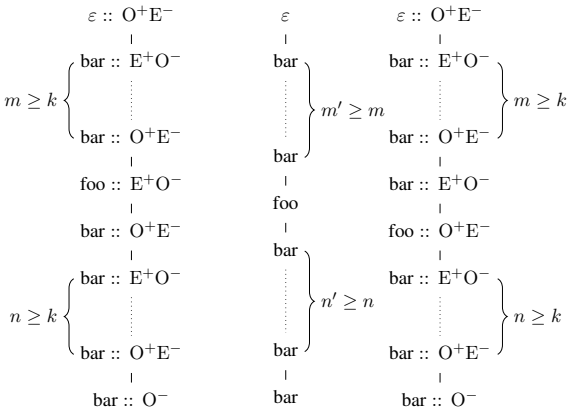

Figure 3: Modulo systems are not ISL-k recoverable

Note first that the value of k can vary depending on other assumptions. MG derivation trees like the one in Fig. 1 display a greater distance between heads and arguments than MG dependency trees like the one in Fig. 2, so the latter will minimize the value for k. This does not mean that the latter is linguistically preferable, but rather that k cannot be fixed independently of the choice of representation. The value of k will also depend greatly on the shape of the phonetic exponents. Fully inflected forms can provide crucial clues about a lexical item's category that would be missing from the uninflected roots postulated in Distributed morphology (Halle and Marantz, 1993). It remains to be seen which set of assumptions and parameters will prove most insightful.

At this point, though, I put forward a maximally restrictive conjecture. Based on a preliminary survey of English data and the linguistic *bon mot* that heads do not select for arguments of arguments, I contend that the category systems of natural languages are maximally simple:

(3) **Complexity of category systems**
Given MG dependency trees with uninflected roots as exponents (e.g. $\sqrt{\text{destroy}}$, $\sqrt{\text{water}}$), it holds for every natural language that all its category and selector features are ISL-2 recoverable.

The conjecture in (3) predicts that whenever a lexical item is categorially ambiguous, its category feature can be determined by inspecting the selecting head or the heads of the selected arguments. Even if ISL-2 recoverability ultimately turns out to be too strong an assumption, ISL-k recoverability still rules out many undesirable feature systems

2.5 Some Linguistic Implications

ISL recoverability correctly rules out some of the most egregious patterns and constraints. But we can try to further limit feature systems based on the size of contexts. Instead of ISL recoverability, the relevant restriction would be ISL-k recoverability for some small k.

and reins in feature coding while allowing for limited categorial ambiguity.

ISL recoverability also has some more indirect consequences. One prediction is that no natural language can have an arbitrarily long sequence $x_1, \ldots, x_n$ such that I) each x_i is an empty head, and II) x_i selects x_{i+1} and nothing else ($1 \leq i < n$). This prediction follows from the fact that unpronounced lexical items provide no overt clues about their category. If the local context does not furnish any pronounced material, local category inference hinges on structural differences. Since the configuration above is structurally uniform, there is insufficient information to correctly infer the categories of all empty heads. This case is interesting because of the proliferation of empty heads in Minimalist syntax. If there is any clear counterexample to (3), it is likely to involve empty heads.

It should also be noted that ISL recoverability is only expected to hold for category and selector features. Features that participate in long-distance dependencies like movement cannot be reliably assigned by an ISL transduction.[2] Consider once more our topicalization example from before. Whether *water* should receive a top^+ feature to license topicalization depends on whether there is some head with a matching top^- feature to undergo topicalization. In the case at hand, this can be made based on the local context alone. But in general, a mover can be arbitrarily far away from its target site, as in *this author, John thinks that Bill said that Mary really adores*. Correct assignment of top^+ thus requires a context of unbounded size, which is impossible with ISL transductions.

Many empirical and theoretical issues remain to be settled. The MG corpus of Torr (2017) may provide valuable clues about the feasibility of conjecture (3), but it must be supplemented by a broad range of typological data. On the formal side, studying the recoverability of movement features will require more powerful extensions of ISL tree transductions. The next section fully formalizes ISL transductions to provide a suitable vantage point for this future work.

[2] In grammars with adjunction, subcategorization can also become a long-distance dependency depending on one's choice of representation (Graf, 2018). A modified version of (3) would predict that the uninflected root of each adjunct still provides enough information to reliably infer category and selector features. I am much more skeptical that this will turn out to be true across all languages.

3 Formal Definitions

This section puts the informal discussion of the preceding section on a formal footing by defining ISL recoverability in terms of *ISL tree relabelings*. But in order to simplify future work on feature recoverability, I define the more general class of ISL tree transductions, which ISL tree relabelings are a particular simple subtype of. The definition of ISL tree transductions differs markedly from that of other tree transductions. Building on Gorn domains and tree contexts (§3.1), I define an ISL tree transducer as a finite set of triples, each one of which maps a node n to a tree context based on the configuration n appears in. The ISL transduction then combines all these tree contexts to yield the final output tree (§3.2). Given this formal apparatus, feature recoverability is easy to state in rigorous terms (§3.3).

3.1 Technical Preliminaries

We define trees as finite, labeled Gorn domains (Gorn, 1967). First note, though, that we use $\mathbb{N}$ to denote the set of all positive natural numbers, i.e. $\{1, 2, 3, \ldots\}$ rather than $\{0, 1, 2, 3, \ldots\}$ — this is non-standard, but will slightly simplify the usage of indices in the definition of ISL-k transducers.

A *Gorn domain* D is a set of strings drawn from $\mathbb{N}^*$, which are called *(Gorn) addresses*, or simply *nodes*. Every Gorn domain must satisfy two closure properties: for all $u \in \mathbb{N}^*$ and $1 \leq i \leq j$ it holds that $uj \in D$ implies both $u \in D$ and $ui \in D$. This entails the inclusion of the empty string ε, which denotes the root. Addresses are interpreted such that u immediately dominates each ui, and each ui is the immediate left sibling of $u(i+1)$.

A Σ-*tree* is a pair $t := \langle D, \ell \rangle$ where D is a finite Gorn domain and $\ell : D \to \Sigma$ is a total function that maps each address to its label, i.e. a member of the alphabet Σ. The *depth* of t is equivalent to the length of the longest Gorn address.

A (Σ, n)-*context* is a Σ-tree whose leaf nodes may also have labels drawn from the set $\{\Box_1, \ldots, \Box_n\}$ of *ports*, which must be disjoint from Σ. Suppose we are given a (Σ, n)-context $c := \langle D_c, \ell_c \rangle$ with $m \leq n$ ports labeled $\Box_i$ at addresses a_1, …, a_m, as well as a tree (or context) $s := \langle D_s, \ell_s \rangle$. Then we use $c[\Box_i \leftarrow s]$ to denote the result of substituting s for each $\Box_i$ in c. This is a new tree $t := \langle D, \ell \rangle$ such that

- $D := D_c \cup \{a_j d \mid 1 \leq j \leq m, d \in D_s\}$, and

- for every $b \in D$

$$\ell(b) := \begin{cases} \ell_s(d) & \text{if } b = a_j d \\ & (1 \leq j \leq m,\ d \in D_s) \\ \ell_c(b) & \text{otherwise} \end{cases}$$

The construction also generalizes to multiple simultaneous substitutions, as in $c[\Box_i \leftarrow s, \Box_j \leftarrow t]$. If c contains no node labeled $\Box_i$, then $c[\Box_i \leftarrow s, \Box_j \leftarrow t] = c[\Box_j \leftarrow t]$ (and $c[\,] = c$).

If S is a set, then substitution can apply in two ways. With *synchronous substitution*, $t[\Box_i \leftarrow S] := \{t[\Box_i \leftarrow s] \mid s \in S\}$. *Asynchronous substitution*, denoted $t[\Box_i \leftarrowtail S]$, yields $\{t[\Box_{i_1} \leftarrow s_1, \ldots, \Box_{i_n} \leftarrow s_n] \mid s_1, \ldots, s_n \in S\}$, assuming that t contains exactly n occurrences of $\Box_i$. Substitution with sets and multiple simultaneous substitutions will be crucial for ISL transductions.

3.2 ISL Transductions

Chandlee (2014) defines ISL string-to-string mappings in terms of deterministic, finite-state string-to-string transducers. Even though the definition does not provide an explicit look-ahead component, ISL mappings can emulate finitely bounded look-ahead via a delayed-output strategy. Suppose, for instance, that a is rewritten as b before d, as c before e, and just as a before f. This is emulated by deleting a and rewriting the next symbol as either bd, ce, or af. Later works define ISL functions in terms of local contexts (not to be confused with (Σ, n)-contexts), and those definitions make look-ahead a standard component to simplify practical work (Chandlee and Heinz, 2018; Graf and Mayer, 2018; De Santo and Graf, 2019).

With tree transducers, the emulation of finitely bounded look-ahead is a much more complex affair that depends on various parameters such as directionality (top-down or bottom-up), totality, and determinism. For this reason, I explicitly add finite look-ahead in the subsequent definitions. I will also allow for non-determinism as future work may require transductions than can handle optionality (e.g. whether a node should receive a movement feature to undergo topicalization).

For the sake of generality and as a starting point for future work, I first define a version of ISL tree transductions that allows for non-determinism, deletion, and copying, and that can run in two different modes of operation (*synchronous* or *asynchronous*). This is subsequently limited to the special case of ISL relabelings, which are the formal core of feature recoverability.

Definition 1 (ISL tree transducer). For any $k \geq 1$, an *ISL-k tree transducer* from Σ-trees to Ω-trees is a finite set τ of *ISL-k rewrite rules* $\langle s, a, t\rangle$, where

- s is a Σ-tree of depth $i < k$,
- a is a node (i.e. a Gorn address) of s with $d \geq 0$ daughters,
- and t is an (Ω, d)-context. ⌟

Definition 2 (Synchronous ISL transduction). The transduction realized by an ISL-k transducer in *synchronous mode* is defined in a recursive fashion. First, a node b in tree u can be *targeted* by an ISL-k context $\langle s, a, t\rangle$ iff there is some $p \in \mathbb{N}^*$ such that

node match $b = pa$, and

label match for all nodes g of s, $\ell_s(g) = \ell_u(pg)$,

full-width match for all nodes gi of s with $g \in \mathbb{N}^*$ and $i \in \mathbb{N}$, if pgj is a node of u ($j > i$), then gj is a node of s.

Now suppose furthermore that n in u has $d \geq 0$ daughters. Given an ISL-k tree transducer τ, we use $\overleftarrow{\tau}(u, b)$ to denote the set of all trees $t[\Box_1 \leftarrow \overleftarrow{\tau}(u, b1), \ldots, \Box_d \leftarrow \overleftarrow{\tau}(u, bd)]$ such that there is a rewrite rule $\langle s, a, t\rangle$ in τ that targets node b in u. If this set is empty, $\overleftarrow{\tau}(u, b)$ is undefined. For any Σ-tree t, we may simply write $\overleftarrow{\tau}(t)$ instead of $\overleftarrow{\tau}(t, \varepsilon)$. For any tree language L, the transduction computed by τ in *synchronous mode* is $\overleftarrow{\tau}(L) := \{\langle i, o\rangle \mid i \in L, o \in \overleftarrow{\tau}(i)\}$. A transduction is *synchronous input strictly k-local* (sISL-k) iff it can be computed by some ISL-k transducer in synchronous mode. It is *synchronous input strictly local* (sISL) iff it is sISL-k for some $k \geq 1$. ⌟

The definition of *asynchronous input strictly k-local* (aISL-k) transductions is exactly the same, except that $\overleftarrow{\tau}$ is replaced by $\overleftarrowtail{\tau}$ such that $\overleftarrowtail{\tau}(u, b)$ denotes the set $t[\Box_1 \leftarrowtail \overleftarrowtail{\tau}(u, b1), \ldots, \Box_d \leftarrowtail \overleftarrowtail{\tau}(u, bd)]$. ISL is used as a shorthand for sISL or aISL, ignoring transduction mode.

The definition of ISL transductions differs from that of other tree transductions in that the input tree is not altered incrementally to yield the output

tree. Instead, each node in the input contributes a context to the output, or rather, a range of possible contexts in the case of a non-deterministic transduction. The transduction then stitches these contexts together in order to arrive at a single tree structure. This stitching is accomplished by the recursive step of mapping $\tau(u, b)$ to $t[\Box_1 \leftarrow \tau(u, b1), \ldots, \Box_d \leftarrow \tau(u, bd)]$. Each $\tau(u, bi)$ $(1 \leq i \leq d)$ corresponds to a context produced from the i-th daughter of the node b in u, and these contexts are inserted into the appropriate ports of the context t produced from n. If n is a leaf node, its output structure is a tree instead of a context. This ensures that the recursion step terminates eventually.

Example. Figure 4 specifies a fragment of an ISL-3 transducer for translating multiplication trees to addition trees (assuming no numbers larger than 3). For simplicity, the ISL rewrite rules are written in a context-free format with a box around the node to be rewritten. An underscore is used to match any arbitrary node label. On the right, a particular input-output mapping is shown with the transducer running in asynchronous mode. In synchronous mode, all $\Box_i$ in the output of rule G would have to be replaced by the same tree. ⌟

It is easy to see that every ISL-k string transduction is an ISL-k tree transduction over unary branching trees. This shows that ISL-k tree transducers are a natural generalization of ISL for strings. However, the current definition goes far beyond ISL string mappings in that it allows for non-determinism and copying.

Definition 3 (Transducer subtypes). An ISL-k tree transducer τ is

deterministic iff it holds for every Σ-tree u that no node of u can be targeted by more than one context of τ,

linear/non-deleting iff all contexts $\langle s, a, t\rangle$ of τ are such that if the node at address a in s has $d \geq 1$ daughters, then t contains every port $\Box_i$ at most once/at least once $(1 \leq i \leq d)$,

structure preserving iff all rewrite rules $\langle s, a, t\rangle$ of τ are such that t is of the form $\omega(\Box_1, \ldots, \Box_d)$ $(\omega \in \Omega)$.

A deterministic, structure preserving ISL-k transducer is called an *ISL-k relabeling*. ⌟

A structure preserving ISL transducer never changes the structure of the input tree. Structure preservation thus entails linearity, which is why the latter is not mentioned in the definition of relabelings. Linearity in turn removes the distinction between synchronous and asynchronous mode as no $\Box_i$ ever has more than one occurrence. Only this very limited type of ISL transducers is relevant for feature recoverability.

3.3 ISL Feature Recoverability

We are finally in a position to define the notion of ISL recoverability that was informally discussed in §2.4. In order to clearly separate features from other parts of the alphabet, we have to track them in a separate component. MGs make this split fully explicit, with a lexical item's phonetic exponent a member of Σ and their feature annotation a string over an entirely separate set of features. Other formalisms such as TAG or GPSG can also be recast along these lines.

Definition 4. Let F be a set of features. An *F-annotated Σ-tree* is a tree whose labels are drawn from $\Sigma \times F^*$. ⌟

Definition 5. Let F be a set of features and e a function that maps each $\langle \sigma, f\rangle \in \Sigma \times F^*$ to σ. Then F is *ISL-k recoverable* with respect to language L of F-annotated Σ-trees iff there is an ISL-k transducer τ such that $\tau(e(t)) = t$ for all $t \in L$.⌟

Note that feature recoverability can vary depending on the particulars of the tree languages. A feature that may not be recoverable with respect to L may be recoverable with respect to L'. Consider once more the grammar in §2.1. If *foo* always had to carry O^-, and *bar* always had to carry E^-, then those category features would be recoverable even though they still encode an even/odd alternation. In this hypothetical scenario, the alternation is tied to overt exponents, reducing *modulo* counting to a strictly 2-local alternation of lexical items. In the other direction, even the simplest (non-trivial) category system cannot be recovered from a language where all lexical items are unpronounced. And as a reviewer correctly points out, if one puts no restrictions on the use of empty heads, features can be encoded in terms of specific structural configurations with empty heads. Feature recoverability thus is a fluid notion that depends equally on the nature of Σ, the syntactic assumptions about structure and phonetic exponents, and the overall complexity of the tree language.

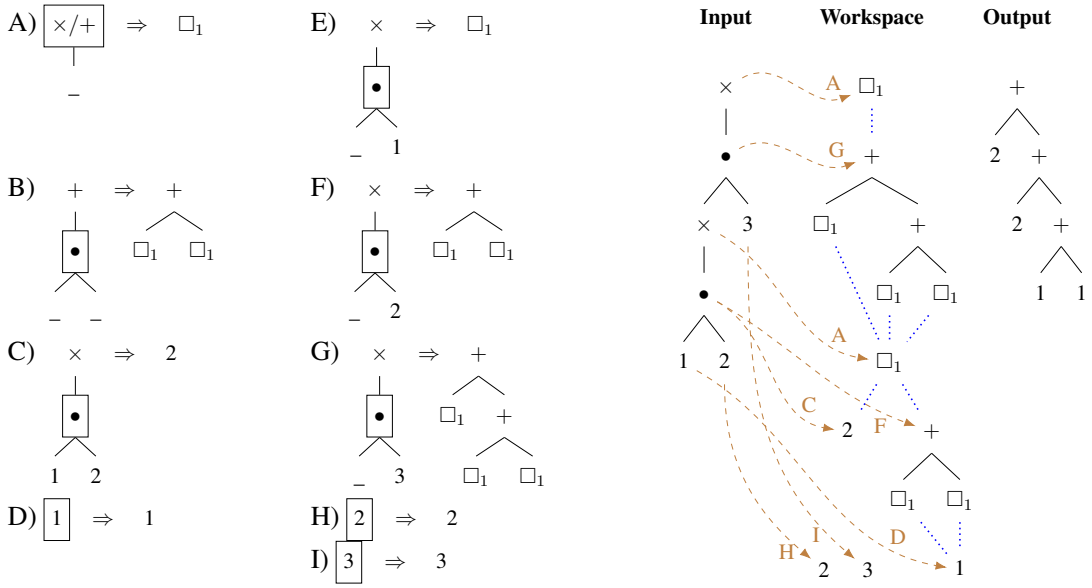

Figure 4: A non-deterministic ISL-3 tree transducer (left) for converting a tree with addition and multiplication to addition only (right). The workspace depicts I) how each node is rewritten as one or more contexts, and II) all possible options for combining these contexts into a particular output tree via substitution steps that are based on the structure of the input tree. Dashed arrows are annotated with the corresponding rewrite rules. The transducer is assumed to operate in asynchronous mode. The output displays one out of $2^3 = 8$ options that differ in when and where rewrite rules C and F are used. In synchronous mode, there would be only two distinct outputs.

4 Conclusion

ISL tree transductions — or more precisely, ISL tree relabelings — offer a reasonable approximation of the limits of category systems in natural languages. I conjecture that all natural languages are such that the category of a lexical item can be inferred from its local context in a tree without any feature annotations. In combination with standard assumptions about linguistic structure, feature recoverability is a powerful restriction that eliminates many of the undesirable cases of feature coding identified in Graf (2017). It also makes strong empirical predictions that merit further investigation by linguists.

Many questions had to remain open. On the formal side, this includes abstract characterizations as well as core properties of ISL tree transductions, e.g. (non-)closure under intersection, union, and composition. The relations to other transduction classes are largely unknown. I conjecture that (deterministic) synchronous/asynchronous ISL transductions are subsumed by (deterministic) bottom-up/top-down transductions with finite look-ahead. Linear ISL transductions should be subsumed by both. The movement features of MGs will require a more powerful kind of transduction, possibly based on the string class TSL (Heinz et al., 2011). There also seems to be a deep connection between feature recoverability and the notion of inessential features (Kracht, 1997; Tiede, 2008).

From a linguistic perspective, one pressing question is to what extent feature recoverability depends on whether syntax uses fully inflected lexical forms or underspecified roots. If fully inflected lexical items do not reduce the complexity of the ISL transduction, or allows for unnatural constraints that would not be possible otherwise, that would be a powerful argument that syntax indeed has no need for anything beyond simple roots.

Acknowledgments

The work reported in this paper was supported by the National Science Foundation under Grant No. BCS-1845344. This paper benefited greatly from the feedback of Jeffrey Heinz, Dakotah Lambert, and three anonymous reviewers. I am indebted to the participants of the University of Tromsø's workshop *Thirty Million Theories of Syntactic Features*, which lit the initial spark that grew into the ideas reported here.

References

Hyunah Baek. 2018. Computational representation of unbounded stress: Tiers with structural features. In *Proceedings of CLS 53*, pages 13–24.

Jane Chandlee. 2014. *Strictly Local Phonological Processes*. Ph.D. thesis, University of Delaware.

Jane Chandlee and Jeffrey Heinz. 2018. Strict locality and phonological maps. *Linguistic Inquiry*, 49:23–60.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Aniello De Santo and Thomas Graf. 2019. Structure sensitive tier projection: Applications and formal properties. In *Formal Grammar*, pages 35–50, Berlin, Heidelberg. Springer.

Saul Gorn. 1967. Explicit definitions and linguistic dominoes. In *Systems and Computer Science, Proceedings of the Conference held at University of Western Ontario, 1965*, Toronto. University of Toronto Press.

Thomas Graf. 2011. Closure properties of Minimalist derivation tree languages. In *LACL 2011*, volume 6736 of *Lecture Notes in Artificial Intelligence*, pages 96–111, Heidelberg. Springer.

Thomas Graf. 2017. A computational guide to the dichotomy of features and constraints. *Glossa*, 2:1–36.

Thomas Graf. 2018. Why movement comes for free once you have adjunction. In *Proceedings of CLS 53*, pages 117–136.

Thomas Graf and Aniello De Santo. 2019. Sensing tree automata as a model of syntactic dependencies. In *Proceedings of the 16th Meeting on the Mathematics of Language*, pages 12–26, Toronto, Canada. Association for Computational Linguistics.

Thomas Graf and Connor Mayer. 2018. Sanskrit n-retroflexion is input-output tier-based strictly local. In *Proceedings of SIGMORPHON 2018*, pages 151–160.

Thomas Graf and Nazila Shafiei. 2019. C-command dependencies as TSL string constraints. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 205–215.

Morris Halle and Alec Marantz. 1993. Distributed morphology and the pieces of inflection. In Ken Hale and Samuel J. Keyser, editors, *The view from building 20*, pages 111–176. MIT Press, Cambridge, MA.

Jeffrey Heinz. 2018. The computational nature of phonological generalizations. In Larry Hyman and Frank Plank, editors, *Phonological Typology*, Phonetics and Phonology, chapter 5, pages 126–195. Mouton De Gruyter.

Jeffrey Heinz, Chetan Rawal, and Herbert G. Tanner. 2011. Tier-based strictly local constraints in phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 58–64.

Gregory M. Kobele. 2011. Minimalist tree languages are closed under intersection with recognizable tree languages. In *LACL 2011*, volume 6736 of *Lecture Notes in Artificial Intelligence*, pages 129–144.

Marcus Kracht. 1997. Inessential features. In Alain Lecomte, F. Lamarche, and G. Perrier, editors, *Logical Aspects of Computational Linguistics*, pages 43–62. Springer, Berlin.

Lei Liu. 2018. Minimalist parsing of heavy NP shift. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation (PACLIC 32)*. Association for Computational Linguistics.

James Rogers. 1997. Strict LT_2 : Regular :: Local : Recognizable. In *Logical Aspects of Computational Linguistics: First International Conference, LACL '96 (Selected Papers)*, volume 1328 of *Lectures Notes in Computer Science/Lectures Notes in Artificial Intelligence*, pages 366–385. Springer.

Edward P. Stabler. 1997. Derivational Minimalism. In Christian Retoré, editor, *Logical Aspects of Computational Linguistics*, volume 1328 of *Lecture Notes in Computer Science*, pages 68–95. Springer, Berlin.

Edward P. Stabler. 2011. Computational perspectives on Minimalism. In Cedric Boeckx, editor, *Oxford Handbook of Linguistic Minimalism*, pages 617–643. Oxford University Press, Oxford.

Hans-Jörg Tiede. 2008. Inessential features, ineliminable features, and modal logics for model theoretic syntax. *Journal of Logic, Language and Information*, 17:217–227.

John Torr. 2017. Autobank: a semi-automatic annotation tool for developing deep Minimalist grammar treebanks. In *Proceedings of the Demonstrations at the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 81–86.

Mai Ha Vu. 2018. Towards a formal description of NPI-licensing patterns. In *Proceedings of the Society for Computation in Linguistics*, volume 1, pages 154–163.

Mai Ha Vu, Nazila Shafiei, and Thomas Graf. 2019. Case assignment in TSL syntax: A case study. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 267–276.