Unbounded recursion in two dimensions, where syntax and prosody meet

Edward P. Stabler
University of California, Los Angeles
stabler@ucla.edu

Kristine M. Yu
University of Massachusetts Amherst
krisyu@linguist.umass.edu

Abstract

Both syntax and prosody seem to require structures with unbounded branching, something that is not immediately provided by multiple context free grammars or other equivalently expressive formalisms. That extension is easy, and does not disrupt an appealing model of prosody/syntax interaction. Rather than computing prosodic and syntactic structures independently and then selecting optimally corresponding pairs, prosodic structures can be computed directly from the syntax, eliminating alignment issues and the need for bracket-insertion or other ad hoc devices. To illustrate, a simple model of prosodically-defined Irish pronoun displacement is briefly compared to previous proposals.

Since phonological structures do not show a principled bound on length, those structures must allow unbounded branching or unbounded depth or both. There is significant controversy about how the balance is struck (Selkirk, 1996, 2011; Ito and Mester, 2012). Idsardi (2018) suggests that the issue can be largely set aside if the appearance of phonological structure derives entirely from the syntax, with a transduction that concatenates segmental material and inserts 'boundary symbols'. But Yu (2021) points out that boundary symbol insertion should not be accidental, stipulated; if there are no prosodic constituents, then we need another explanation of 'boundary' distribution. Rigorous studies of these matters are often based on grammars and automata that do not provide mechanisms for unbounded branching. This absence may obscure part of our picture of the syntax-prosody interface.

For syntax, Chomsky (1961, 1963, 2018) observes that standard rewrite grammars do not provide unbounded branching:

> The failure of strong generative capacity of [phrase structure grammar] . . . is a failure of principle, as shown by unstructured coordination: e.g., "the man was old, tired, tall,. . . , but friendly". Even unrestricted rewriting systems fail to provide such structures, which would require an infinite number of rules. The more serious failure, however, is in terms of explanatory adequacy. (Chomsky, 2018, p.132)

Chomsky's remarks about this are discussed in Lasnik (2011) and Lasnik and Uriagereka (2022, pp.15-20). Lasnik (2011) notes that Chomsky and Miller (1963, p. 298) actually consider this context free rule for adjective coordination:

$$\text{Predicate} \rightarrow \text{Adj}^n \text{ and Adj} \qquad (n \geq 1).$$

However, as Lasnik notes:

> Chomsky and Miller indicate that there are "many difficulties involved in formulating this notion so that descriptive adequacy can be maintained. . . ". But they do not elaborate on this point. It would surely be interesting to explore this. . . (Lasnik, 2011, p.361)

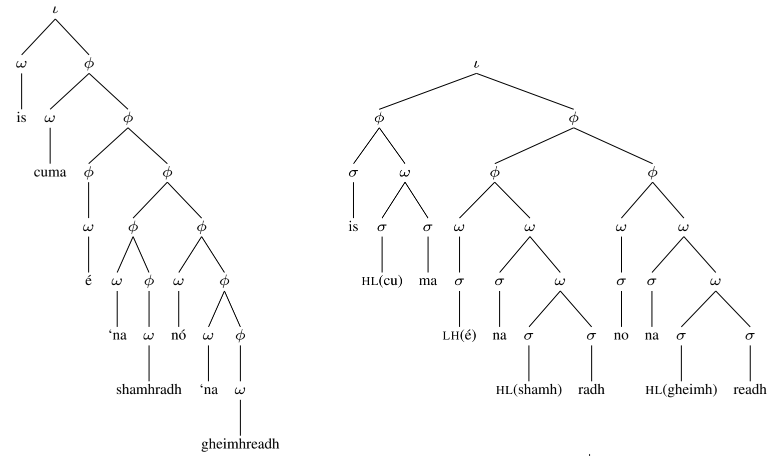
That option is explored here.

Inspired by Kleene (1956), unbounded branching can be added to phrase structure rewrite grammars by allowing the Kleene star $*$ on the right side of any rule.[1] Yu (2022), reviewed in §1, proposes that prosodic constituency and dependencies can be specified by multi bottom up tree transducers or, equivalently, multiple context free grammars. These can also be extended with $*$ on the right side of any rule, accommodating unbounded prosodic branching. In recent syntax too, the evidence supports unbounded branching. Neeleman et al. (2023) defends unbounded branching for coordination, and briefly reviews the long history of such proposals. McInnerney (2022b) argues for unbounded branching in adjunction. And Chomsky (2021, p.20) recently proposes a *-extension of merge, in his rule 'D'.

[1] This idea is used in finite state toolkits (Beesley and Karttunen, 2003; Hulden, 2009; Gorman and Sproat, 2021), and *-extended context free grammars are commonly used to define programming languages (Wirth, 1977; Albert et al., 2001; Martens and Niehren, 2005; Jim and Mandelbaum, 2010; Borsotti et al., 2023). Pattis (1994) argues that context free grammars with unbounded branching should be taught on the first day of your first class in Computer Science.

Proceedings of the Society for Computation in Linguistics (SCiL) 2023, pages 343-356.
Amherst, Massachusetts, June 15-17, 2023

$\iota(xy) \leftarrow \omega(x)\ \phi(y)$
$\phi(xy) \leftarrow \omega(x)\ \phi(y)$
$\phi(xy) \leftarrow \phi(x)\ \phi(y)$
$\phi(x) \leftarrow \omega(x)$
$\omega(xy) \leftarrow \sigma(x)\ \omega(y)$
ω(is) $\leftarrow$
ω(cuma) $\leftarrow$
ω(é) $\leftarrow$
ω(’na) $\leftarrow$
ω(shamhradh) $\leftarrow$
ω(gheimhreadh) $\leftarrow$
ω(nó) $\leftarrow$



a. MCF grammar b. Derivation from grammar a c. Derivation with $\iota(\vec{x}) \leftarrow \phi^+(\vec{x})$ and pitch accents

Figure 1: $*$-MCF prosody for (1)

A $*$-extension of minimalist grammar is proposed in §2, providing an analog of Chomsky’s rule D. And since the primary causal influences in prosody and syntax differ, it is natural to define them separately. But this creates a puzzle about how the respective influences interact in linguistic performance. A syntax-prosody interface inspired by Bennett et al. (2016) is proposed, one that allows prosodically conditioned pronoun postposing of weak pronouns in Irish. These pronouns can appear middle of a coordinate structure, suggesting a non-syntactic placement. With a syntax for Irish coordination that allows unbounded branching, postposing of these weak elements can occur in the generation of prosodic structure.

1 $*$-Prosodic structure

Yu (2021) points out that many phonologists argue for structures with branching constituencies that finite state automata do not provide. And Yu (2022) observes that certain multiple dependencies, sometimes marked with arcs in representations of phonological structure (Pierrehumbert and Beckman, 1988), can be captured and made explicit in ‘finite state multi bottom up tree transducers’ (MBOTs) or, equivalently, in ‘multiple context free grammars’ (MCFGs). A simple MCFG is presented in Figure 1a using the logic-based notation of Kanazawa et al. (2011). Each rule is a conditional, with the back arrow $\leftarrow$ pronounced “if”, and with variables over strings on the right that get concatenated on the left side of each rule. The first rule Figure 1a says “xy is an ι if x is a ω and y is a ϕ”. The last says “the string *nó* is an ω”.

Here, we also extend MCFGs with the Kleene star and plus. Any category C on the right side of a rule can be starred, C^*, meaning that it may occur 0 or more times, where the strings of this sequence are adjacent in the sequence $\vec{x}$. We also allow C^+ which is the same as C followed by C^*. So the rule

$$\iota(\vec{x}) \leftarrow \phi^+(\vec{x})$$

says “the strings of $\vec{x}$, concatenated, are an ι if they are the strings of one or more occurrences of ϕ”. An instance of that rule is applied at the root of Figure 1c. And the rule from Chomsky, mentioned in the introduction, is:

$$\text{Predicate}(\vec{x} \text{ and } y) \rightarrow \text{Adj}^+(\vec{x})\ \text{Adj}(y).$$

That says “$\vec{x}$, concatenated with *and*, concatenated with string y, is a Predicate if $\vec{x}$ are the strings of 1 or more Adj, and y is also an Adj”. We will also write LH(é) to indicate that é has the pitch accent LH, and similarly for HL. These extensions do not change MCFG expressive power (Appendix A). MCFGs are ‘multi’ in allowing categories to classify multiple strings – relevant in §3.

Example. Consider the Irish (1) from Bennett et al. (2016), in which we added syntactic bracketing for the coordinate structure:

(1) is cuma [é [‘na shamhradh] [nó [‘na gheimhreadh]]]
COP.PRES no.matter it PRED summer or PRED winter
‘It doesn’t matter if it’s summer or winter’

Assuming the syntactic structure in Figure 4 (excluding the conjunct *‘na fhómar*), the prosodic structure expected following the syntax-prosody mapping principles of Match Theory (Selkirk, 2011; Elfner, 2012) is shown in Figure 1b. In brief, optimality-theoretic MATCH constraints enforce that clausal projections correspond to intonational phrases (ι), maximal projections to phonological phrases (ϕ), and heads to prosodic words (ω). However, Bennett et al. (2016, (104)) propose that the prosodic structure in fact phrases pronoun é together with the first conjunct *'na shamhradh* in a single ϕ, like in Figure 1c.

Briefly, to explain this, they propose that prosodic markedness constraints are ranked above MATCH constraints, following Elfner (2012, §4.2). The key prosodic markedness constraints are: (i) EQUALSISTERS (Bennett et al., 2016, (48)), which assigns a violation when sisters are not of the same prosodic category (Myrberg, 2013), (ii) STRONGSTART (Bennett et al., 2016, (55)), which penalizes ϕ- and ι-phrases with leftmost daughters that are "prosodically dependent", i.e., syllables (σ), and (iii) BINARITY, which penalizes nodes that are not binary branching. Here we assume that BINARITY is applicable only to ϕ-nodes, following Elfner (2012, §4.2), and that EQUALSISTERS is applicable only to nodes above the prosodic word (since Myrberg (2013) and Bennett et al. (2016) consider only above the level of the prosodic word). In addition, Bennett et al. (2016, p. 198) assume that a prosodic word must contain a stressed syllable, which we can encode as an inviolable CULMINATIVITY constraint.

While the tree in Figure 1b incurs no MATCH constraint violations, it incurs five EQUALSISTERS violations due to $\langle\omega,\phi\rangle$ daughter pairs, as well as three BINARITY violations due to unary branches to *é*, *shamhradh*, and *gheimhreadh*; moreover, *is* and *'na* (but crucially, not *é*) are stressless clitics and thus incur violations of CULMINATIVITY. In contrast, the prosodic tree in Figure 1c incurs a number of MATCH violations, but no BINARITY violations and only single STRONGSTART and EQUALSISTER violations due to the phrasing of the daughters *is* and *cuma*. The structure in Figure 1c with pronoun *é* linearized preceding the conjuncts is only optimal when *é* occurs in its strong, stressed form. When *é* occurs in its weak, unstressed form, it cannot form a prosodic word on its own—only a syllable. If the ω node over *é* in Figure 1b was deleted, leaving just a σ, violations of EQUALSISTERS and STRONGSTART would be incurred.

2 $*$-Minimalist grammar

Minimalist grammars (MGs) are weakly equivalent and closely related to MCFGs (Harkema, 2001a; Michaelis, 2001) and can be similarly extended to unbounded branching, leaving weak expressive power unchanged (Appendix A). Here we adapt the version of MG in Kobele (2021), which has only positive and negative feature occurrences, where expressions are formed by merging expressions in which each negative occurrence is 'mated' with a positive occurrence.

We use only one polarity relation following Kobele (2021) and others.[2] Initially, let a minimalist grammar (MG) be a finite set of lexical items that associate phonological forms with feature-based formulas as follows:

feature	::=	V | D | A | C | wh | ...
		| feature$^+$ |feature* | X
non-empty-conj	::=	feature | feature . non-empty-conj
conj	::=	ϵ | non-empty-conj
formula	::=	conj $\multimap$ non-empty-conj
lexical-item	::=	phonological-form : formula

In any formula, features in the antecedent conjunction on the left are are negative; those in consequents positive. When an antecedent is empty, instead of $\epsilon \multimap$ a.b or $\multimap$ a.b, we often write a.b.

$*$-Merge. We extend the usual definition of binary merge to to allow any number of constituents to be combined in one step:

$$\mathrm{M}(A, B, C_1, \ldots, C_n) = \{A, B, C_1 \ldots, C_n\}.$$

At least 2 constituents are required, so it is sometimes convenient to write $A, B, \vec{C}$ for $A, B, C_1, \ldots, C_n$ $(n \geq 0)$. Sets are unordered, of course, but order would be redundant since, as will become clear, heads and subcategorized elements are distinguishable by their labels.

Labels. Derivations begin with *numerations*, which are defined here as finite sequences of lexical and derived elements. Merge applies to numeration elements, replacing them. And the merge steps of a successful derivation produce complexes which can be assigned a label by function ℓ. A lexical or derived structure A whose first unmated feature is

[2]MGs often use 2 canceling pairs (=x selects x, and +x licenses -x), but here we use 1. A head (negative occurrence of x) 'mates' or 'cancels' a non-head (positive occurrence of x). Eliminating the move/merge distinction arguably makes scope reconstruction less surprising (Sportiche, 2017; Chomsky, 1995, §3.5). Cf. CMGs (Stabler, 2011), e-MGs (Chesi, 2021), and Horn linear logic (Kanovich, 2015).

Labels are defined with 3 cases (lexical items, internal merge, and external merge, respectively):

$$\ell(A) = \begin{cases} A : \{\alpha \multimap \beta\} & \text{if } A \text{ is a lexical item } w : \alpha \multimap \beta \\ A : \gamma & \text{if } A = \{B, C, \vec{D}\},\ C : F \in \ell(B), \gamma = m(\ell(B), \{C : F\}) \text{ is defined, and } \&(\ell(C), \vec{D}) \\ A : \gamma & \text{otherwise, if } A = \{B, C, \vec{D}\}, \gamma = m(\ell(B), \ell(C)) \text{ is defined, and } \&(\ell(C), \vec{D}) \end{cases}$$

Tentatively, $\&(\alpha, \vec{D})$ iff every element of $\vec{D}$ has label α.
And the 'mating' function calculates the labels of complexes, for the third case of ℓ:

$$m(S[f.\alpha \multimap \beta], T[B : \{f.\gamma\}]) = \begin{cases} \{\alpha \multimap \beta\} \cup S \cup T & \text{if } \gamma = \epsilon \text{ and smc}(S \cup T) \\ \{\alpha \multimap \beta, B : \gamma\} \cup S \cup T & \text{if } \gamma \neq \epsilon \text{ and smc}(\{B : \gamma\} \cup S \cup T), \end{cases}$$

where $X[\alpha]$ is a set X containing formula α and then X is the result of removing that element, and where smc(X) iff no two formulas in X have the same first unmated feature.

Figure 2: MG label checking

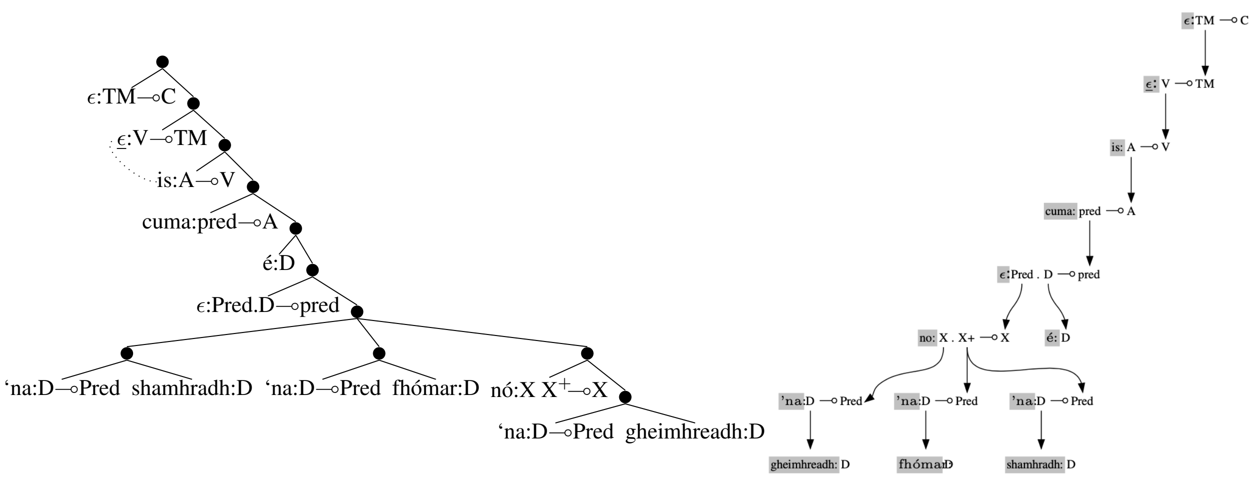


Figure 3: Left, structure for (2): a set in which leaves are lexical items, internal nodes are sets, arcs are $\in$ relations. A dotted arc is added to indicate PF head movement, independent of syntax-derived set. Right, the corresponding dependency tree (with feature-checking arcs).

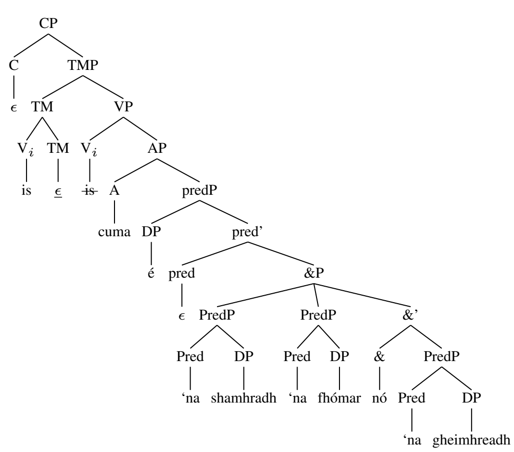


Figure 4: X-bar tree for derivation of Figure 3.

negative f can *mate* with a lexical or derived structure B whose first unmated feature is positive f. Labeling maps a lexical item or derived set A to a pair $A : F$, where F is the set that contains the formula of the head, but with mated feature removed, together with the pairs $\vec{B} : \vec{G}$ of subconstituents $\vec{B}$ with unmated positive features $\vec{G}$, as detailed in Figure 2.

As in previous MGs, ℓ requires embedded positive elements to satisfy the 'shortest move constraint' (smc): $\ell(A, B)$ is undefined if A, B have any first positive feature in common. The mating m then applies to the labels. Writing $N[A, B, \vec{C}]$ when A and $\vec{C}$ are in numeration N and either (i) $\vec{B} \in N$ or (ii) $B \in \ell(A)$, let $N[\mathrm{M}(A, B, \vec{C})]$ be the result of letting $\{A, B, \vec{C}\}$ replace A, B and $\vec{C}$

in N.[3] We call steps using labeling condition (i) *external merge* and steps using (ii) *internal merge* or *move*. Note that a move-over-merge condition is imposed in the definition of the labeling ℓ in Figure 2 – it's the last, 'otherwise' case.[4]

The labeling of pairs A, B is extended to the labels of $A, B, \vec{C}$ by requiring each element of C to have the same label as B, and assigning the complex the same label it would have if $\vec{C}$ were empty. In lexical entries, f^+ is a special feature that allows labeling of $\vec{C}$, with 1 or more elements with first negative feature f. For convenience, in any lexical item, we also allow variable X to be instantiated with any single feature.

Derivations. Now a rule R, building syntactic structures from elements of a numeration, can be formulated like this:

$$\frac{N[A,B,\vec{C}]}{N[\mathrm{M}(A,B,\vec{C})]}\ \text{(R) if } \ell(\mathrm{M}(A,B,\vec{C})) \text{ is defined.}$$

A structure is *complete* when it has exactly one unmated feature, that feature is on its head, and it is positive. And a derivation from numeration is complete when we have derived a single complete structure. The grammar defines the set of complete structures derived from numerations of its elements. For any feature c, let L_c be the set of sets of non-empty phonological forms at the leaves of completed structures with that feature.

Linearization. Unlike rule R, parsers construct derivations from numerations of zero or more non-empty and often ambiguous phonological forms, and linear order matters. For any grammar G define

$$G(x) = \begin{cases} \{A \in G \mid A = x : F\} & \text{if } x \text{ is phonological} \\ \{x\} & \text{otherwise.} \end{cases}$$

Tentatively, let's adopt the Kayne-like idea that first-mated elements are pronounced to the right of the head later-mated elements on the left, with elements pronounced only in their derivationally latest positions.[5]

[3]Appendix C has a complete implementation of R. With compilers that avoid 'destructive' operations, 'replacement' of $A, B, \vec{C}$ by $\{A, B, \vec{C}\}$ need involve no deletion, but rather a change in how the elements are accessed (Wadler, 1992).

[4]Following Kobele (2021). Sometimes merge-over-move is assumed (Epstein et al., 2012; Chomsky, 2000, p.106), but that has been challenged on empirical grounds (Shima, 2000; Castillo et al., 2009; Abels, 2012, §4.3.1). Careful discussion of the these alternatives, and their interaction with the smc and island constraints, is beyond the scope of this brief study.

[5]See e.g. Kayne (2020, 1994); Collins and Kayne (2020); Johnson (2017); Biberauer et al. (2014); Nunes (1999).

Order is further complicated by 'head movement', which we assume is non-syntactic, morphologically-driven (Harizanov and Gribanova, 2019; Chomsky, 2021, i.a.). A morphological feature of a selecting head can attract the head of a selected complement to its left.

Let's call this rule K:

$$\frac{N[x, y, \vec{z}]}{N[\mathrm{M}(A, B, \vec{C})]}\ \text{(K)} \quad \text{if} \quad \begin{array}{l} A \in G(x), B \in G(y), \vec{C} \in G(\vec{z}), \\ \ell(\mathrm{M}(A, B, \vec{C})) \text{ is defined, and} \\ \text{if this is } B\text{'s last mating, then} \\ \quad (\text{ if this is } A\text{'s first mating,} \\ \quad\ \text{then } A, B, \vec{C} \text{ are adjacent in } N; \\ \quad\ \text{else, } \vec{C}, B, A \text{ are adjacent), and} \\ \text{a morphological feature of } \underline{\text{A}} \text{ can attract} \\ \text{the phonetic head of first merged B.} \end{array}$$

A simple model of rule K is implemented by the minimalist grammar mechanisms of Stabler (2001) and Stanojević (2019).[6]

In the long tradition of generalizations about linear precedence, this idea is among the simplest.[7] MGs adopting this idea are very expressive, defining a mildly context sensitive class of languages (Michaelis, 2001; Harkema, 2001b).

Example, continued. Consider this 3-coordinate elaboration of the previous example:

(2)	is	cuma	é	'na	shamhradh,	'na
	COP.PRES	no.matter	it	PRED	summer,	PRED

fhómhar	nó	'na	gheimhreadh
autumn,	or	PRED	winter

'It doesn't matter if it's summer, autumn or winter'

We assume that the head movement shifts the copula from V to a tense-modality position TM below the complementizer C (McCloskey, 2022). And we assume that a predP small clause is the complement of the adjective. Then a structure similar to the one proposed by Bennett et al. can be defined by this lexicon, indicating the morphological feature of the empty head-raising TM by underlining it:

ϵ: TM $\multimap$ C	$\underline{\epsilon}$: V $\multimap$ TM	is: A $\multimap$ V
cuma: pred $\multimap$ A	ϵ:Pred.D $\multimap$ pred	
'na: D $\multimap$ Pred	nó: X $X^+ \multimap$ X	
shamhradh: D	fhómar: D	gheimhreah: D

[6]A further extension is proposed for coordinate structures by Torr and Stabler (2016): when all coordinates have the same head, they can all be 'adjacent' to the selecting head in the sense required for head movement in (K). And note that Figure 2's requirement that coordinates have identical types is too strong. Relaxing that condition to handle ellipsis, etc., the higher order structures of type logics are valuable (Kubota and Levine, 2021, and references cited there). Even in that powerful system, it is not yet clear how to avoid lexical redundancies and other issues (Morrill and Valentín, 2017). Kobele (2019) extends a minimalist grammar with similarly higher-order structures, but further exploration of these issues is left for future work.

[7]Cf. e.g. Shieber (1984); Daniels and Meurers (2004); Abels and Neeleman (2012); Cinque (2017); Kusmer (2020); Stanojević and Steedman (2021); Roberts (2021).

From any numeration that contains exactly 1 occurrence of each of these elements, we can derive the complete structure depicted by Figure 3 left, where internal nodes are sets with downward arcs to their respective elements. Figure 3 also shows the corresponding dependency graph, and Figure 4 an X-bar structure.[8] Clearly, with numerations of elements from this 10 element lexicon, we can derive not only (2) but also (1) and an infinite number of other structures of category C, with any number of coordinates.

3 The meeting point

Bennett et al. (2016) note that there are variants of (1) in which pronoun é is prosodically weak and postposed, with prosodic structures shown in Figure 5: [9]

(3) is cuma 'na shamhradh é nó 'na gheimhreadh
COP.PRES no.matter PRED summer it or PRED winter

(4) is cuma 'na shamhradh nó 'na gheimhreadh é
COP.PRES no.matter PRED summer or PRED winter it

For a syntactician, (3) is a puzzle. Why and how could a pronoun be displaced into the middle of a coordinate structure? Bennett et al. suggest that this happens for reasons that were already needed in the account of (1). Because the pronoun é is prosodically weak, it doesn't adjoin at the left edge of the first conjunct in (1) like in Figure 1c, where it would incur both STRONGSTART and EQUALSISTER violations. Instead, it avoids violating STRONGSTART via postposing. In fact, the Bennett et al. OT account of (1) extends almost immediately to (3) and (4) once we allow the prosody to consider candidates with displacement. Here we show that proposal has a transparent and efficient computational implementation.

A common idea is that the relation GEN pairs each syntactic structure input with all possible prosodic trees, or all prosodic trees that yield the same string of pronounced elements. Then MATCH can require that each syntactic XP correspond to a ϕ in the prosodic structure. But the number of possible trees can be very large, and how are corresponding (XP,ϕ) pairs found? Counting each XP and requiring a corresponding number of ϕ is unnecessarily nonlocal and inefficent. Requiring that each XP have an ϕ dominating the same words is worse – many XPs can have the same words, so how do we keep track of them?

A natural idea is to represent the set of candidates for any input with a finite state transducer. A tree transducer is simply a device that traverses an input tree, going into one of finitely many states at each point. Bottom-up transducers traverse the input from the leaves up to the root. Traversing the input, the output tree is extended in each step by rules that depend on the current state and the next symbol of the input tree. A transducer that is 'multi' has states that can have several output subtrees at once, allowing it to move things up through the tree, to be assembled into the structure later. We also allow our transducers to be 'extended', which means that a rule can look at more than just one symbol of the input at a time, allowing simpler rules. So we use XMBOTs, finite state extended multi bottom up tree transducers (Engelfriet et al., 2009).

In a transduction from an input to an output tree, an alignment is established by the operation of the transduction itself. Traversing an input XP, the transducer will either output the corresponding ϕ or not, and the latter case can be penalized. And more generally, when all the constraints are themselves definable by finite state transducers, an important result from string-based OT carries over to the setting: a guarantee that optimal structures can be computed efficiently (Ellison, 1994; Eisner, 1997; Albro, 1997; Heinz et al., 2009).[10] In this setting, instead of considering each candidate one-by-one, we apply constraints to the finite state grammar that generates all the candidates. Large candidate sets are then unproblematic, so we can allow candidates with displacement, and candidates that skip levels in the prosodic hierarchy.[11]

[8]Standard sets related by membership are multidominance structures, but they are simpler than some multidominance structures of earlier proposals (Gärtner, 2002, 2014; Citko, 2011). MG dependency graphs are used by Kobele (2021), Salvati (2011), Stabler (1999), inspired by proof nets (Moot and Retoré, 2012; Moot, 2002; Girard, 1987). And for computing X-bar structure, see e.g. Stabler (2013, App.B).

[9]Cf. Chung and McCloskey (1987); McCloskey (1999); Duffield (1995); Adger (1997, 2007); Mulkern (2003, 2009); Elfner (2012); Bennett et al. (2016); Windsor et al. (2018); Kusmer (2020).

[10]See Daland (2014) and Heinz and Idsardi (2017) for brief comparison of this computational model with others prominent in phonology.

[11]This tree-based strategy, expressing GEN and constraints with composable finite state transducers, was suggested by Graf (2012a,b), and is the natural option here. In contrast, Kalivoda (2018, (179)), Bellik and Kalivoda (2017, Appendix) and Kalivoda and Bellik (2020, §4) define GEN as a set of

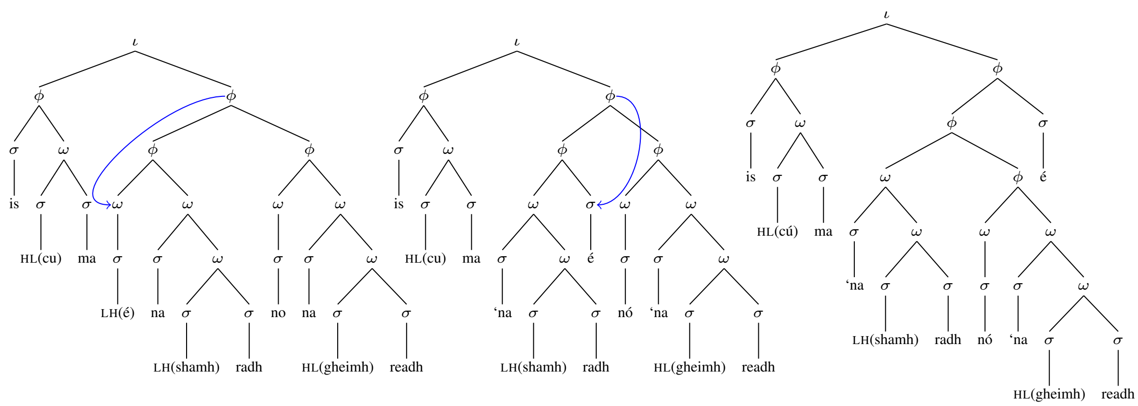
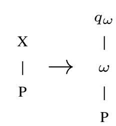


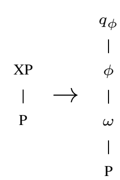
Figure 5: Prosody for (1), (3), (4): attaching é left of sister's 1st daughter, right of that daughter, and right of sister

Engelfriet et al. (2009) point out that MCFGs are just MBOTs that compute string yields, and so the $*$-extension of XMBOTs is similar. And for any two linear XMBOTs, we can construct a single XMBOT that computes their composition. When GEN is an XMBOT, and each constraint is an XMBOT that marks some structures every time the constraint is violated, then we can compose GEN with the top-ranked constraint for an XMBOT that still generates all candidates but with additional marks on the steps that violate constraints. Then, using Dijkstra's algorithm, paths that produce more constraint violations than necessary can be pruned to generate only structures that are optimal with respect to that first constraint. Iterating this step to apply constraints from the most highly ranked to the lowest, pruning suboptimal paths in each result, the algorithm stops when there is only one remaining candidate or when all constraints have been evaluated. This exactly simulates a tableau evaluation, and is guaranteed to be efficient even when the candidate sets are large or infinite.[12]

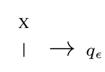
For illustration, let's take a few steps in the derivation of a prosodic structure, beginning with the familiar X-bar structure in Figure 4, except, as in §1, we leave out indices and the middle coordinate. For this example, we use 4 states $q_\omega, q_\phi, q_\iota, q_\epsilon$, with q_ι the final state. For nonempty head category X (that is, for V,A,Pred,&) with phonetic content P, we have the rule:



For phrasal category XP with phonetic content P:

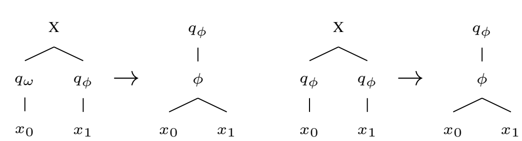


For any category X:

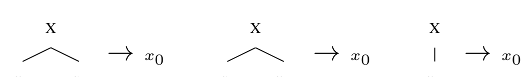


That set of rules, applied bottom up, replaces all the terminal elements of Figure 4 by states with subtrees.

For internal nodes, variables x_0, x_1 range over subtrees. For non-head categories X,



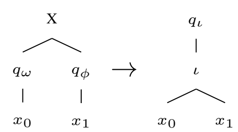
And for any category X



pairs. They require that the order of pronounced elements in the input and output are the same, so prosodic displacements are not among the candidates. Bellik et al. (2021, fn3) clarifies that their trees also do not include level-skipping, apparently disallowing e.g. ϕ parents of σ in Figure 1b,c. Kusmer (2020, §6.1) defines GEN to allow the (much larger) set of pairs in which all orders of pronounced elements appear among the output candidates, and does not confront the computational problem. Dolatian et al. (2021) does propose using a transducer to map from syntax to prosody, but does not use OT.

[12] Frank and Satta (1998) credit Paul Smolensky with noting that this kind of approach, with a pruning step that does not require any finite bound on violations, can be non-finite-state, unlike e.g. 'lenient composition' (Karttunen, 1998). A referee conjectures that our constraints are 'global' (Jäger, 2002), guaranteeing finite-stateness. And other regular versions of OT might extend naturally to prosodic trees, e.g. Lamont (2022). We leave these broader issues for later work.

Finally, we add a 9th rule:



These rules suffice to map the X-bar tree to the prosodic structure in Figure 1a, along with many other candidate structures. (See Appendix C.)

To guarantee closure under composition, note that these rules are linear in the sense that each variable on the left appears at most once on the right. And note that the rules are nondeterministic, because the left side of the last rule – a rule for ι – is identical to the left side of one of the rules for ϕ. Among the properties of these rules that are linguistically important: phonetically empty structure is discarded; and MATCH-governed alignments are completely transparent. That is, rules that process heads but do not introduce an ω are violating, as are rules processing XP without introducing a ϕ, and rules that process clauses without introducing ι. And of course we can track alignments in more complex rule sets where the alignments are not quite so transparent.

Figure 1b is good for MATCH, but violates other constraints that may be ranked more highly, like BINARITY. We can easily see which rules create non-binary structures. So if, for a given input, it is possible to avoid those rules, we can throw them out – the algorithm informally described above automates the discovery of such non-optimal offenders. More importantly, XMBOTs, because they are 'multi', can move also things around. That is, in effect, they can delay the construction of the ϕ dominating the conjuncts in the structures of Figure 5 until the pronoun comes into view. This allows the more optimal, displaced alternatives in the middle and right trees of Figure 5 to be constructed when é is weak, since these alternatives are available.

All the constraints mentioned in the §1 sketch of the Bennett et al. (2016) proposal can be defined as XMBOTs. So efficient computation of optimal prosody from $*$-MG derivations is guaranteed.[13]

[13] Dolatian et al. (2021) points out that the stress rule proposed for coordinate structures by Wagner (2010) is not computed by any XMBOT. The empirical basis of Wagner's proposal could be challenged, or, as Dolation et al. speculate, Wagner's stress rule could be implemented by allowing a very restricted copying. We leave this for future work.

4 Parsing and future work

Seki et al. (1991) present an MCFG parsing algorithm that is succinctly reviewed by Kallmeyer (2010, §7.1), who says "The idea is that once all the predicates in the right side of a rule have been found, we can complete a left side". To allow star and plus categories C^*, C^+ on the right side, there are two cases. Non-empty categories are expand as possible in the chart, exactly as if there were rules with any number of Cs. Empty categories, on the other hand, can introduce cycles in the chart of completed constituents, just as right recursion over empty categories does.

$*$-MGs with Rule K can also be parsed directly. In the bottom-up MG parsing of Harkema (2001b, §4.4), for example, the required adjustment is almost identical to the one for Seki's MCFG parser. Instead of arbitrarily many MCFG rules, Harkema has merge, treated in 5 cases, but the complete rules are essentially the same. So for starred features in a merge rule, any number of constituents is allowed to match. An implementation is linked in fn. 17.

For any MG structure, we compute optimal prosodic structure by $*$-extended transductions, with 'unranked' trees. There are already tree transducer libraries (Bahr, 2012; May and Knight, 2006; Genet and Tong, 2001; Rival and Goubault-Larrecq, 2001), but an up-to-date tree-based toolkit designed specifically for linguists would be useful, analogous to the finite state string toolkits mentioned in fn. 1. This would provide an efficient way to explore a large range of proposals about syntax/phonology interaction, even in cases where large or infinite candidate sets need to be assessed.

Looking at unbounded coordination in Irish also raises linguistic issues that are left for future work. Consulting Irish linguists, it seems, at least to some, that the pronoun in the 3 coordinate case can be initial or final, but nowhere inside the coordinate structure.[14] It seems unlikely that BINARITY should hold in this and longer, list-like coordinations.

More generally, it is not clear that this is the right way for syntax to meet prosody, but the formal model perhaps makes some aspects of the situation clearer. And the $*$-extension of MG syntax should be unified with previous ideas about 'persistent' features (Stabler, 2011; Graf and Kostyszyn, 2021), and with the broader TSL program (Heinz et al., 2011; Graf, 2022).

[14] We are grateful for advice, judgements and references from James McCloskey, Dónall Ó Baoill, and Ryan Bennett.

References

Klaus Abels. 2012. *Phases: An Essay on Cyclicity in Syntax*. Linguistische Arbeiten.

Klaus Abels and Ad Neeleman. 2012. Linear asymmetries and the LCA. *Syntax*, 12(1):25–74.

David Adger. 1997. VSO order and weak pronouns in Goidelic Celtic. *Canadian Journal of Linguistics*, 42(1-2):9–29.

David Adger. 2007. Pronouns postpose at PF. *Linguistic Inquiry*, 38(2):343–349.

Jürgen Albert, Dora Giammarresi, and Derick Wood. 2001. Normal form algorithms for extended context-free grammars. *Theoretical Computer Science*, 267(1–2):35–47.

Daniel M. Albro. 1997. Evaluation, implementation, and extension of primitive optimality theory. Master's thesis, UCLA.

Patrick Bahr. 2012. Modular tree automata. In *Mathematics of Program Construction*, pages 263–299. Springer LNCS 7432. Code: https://hackage.haskell.org/package/compdata-automata.

Kenneth R. Beesley and Laurie Karttunen. 2003. *Finite State Morphology*. CSLI.

Jennifer Bellik, Junko Ito, Nicholas Kalivoda, and Armin Mester. 2021. Matching and alignment. In Haruo Kubozono, Junko Ito, and Armin Mester, editors, *Prosody and Prosodic Interfaces*. Oxford University Press.

Jennifer Bellik and Nicholas Kalivoda. 2017. Syntax-prosody in optimality theory. Technical report, University of California, Santa Cruz.

Ryan Bennett, Emily Elfner, and James McCloskey. 2016. Lightest to the right: An apparently anomalous displacement in Irish. *Linguistic Inquiry*, 47(2):169–234.

Theresa Biberauer, Anders Holmberg, and Ian Roberts. 2014. A syntactic universal and its consequences. *Linguistic Inquiry*, 45(2):169–225.

Angelo Borsotti, Luca Breveglieri, Stefano Crespi Reghizzi, and Angelo Morzenti. 2023. General parsing with regular expression matching. *Journal of Computer Languages*, 74:101176.

Juan Carlos Castillo, John E. Drury, and Kleanthes Grohmann. 2009. Merge over move and the extended projection principle. *Iberia*, 1(1).

Cristiano Chesi. 2021. Expectation-based minimalist grammars. *Computing Research Repository*, arXiv:2109.13871.

Noam Chomsky. 1961. On the notion 'rule of grammar'. In *Structure of Language and its Mathematical Aspects: Procs. 12th Symposium in Applied Mathematics*, pages 6–24.

Noam Chomsky. 1963. Formal properties of grammars. In R. Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology, Volume II*, pages 323–418. Wiley.

Noam Chomsky. 1995. *The Minimalist Program*. MIT Press.

Noam Chomsky. 2000. Minimalist inquiries: The framework. In R. Martin, D. Michaels, and J. Uriagereka, editors, *Step by Step: Essays on Minimalism in Honor of Howard Lasnik*, pages 89–155. MIT Press.

Noam Chomsky. 2018. Syntactic structures. some retrospective comments. In Norbert Hornstein, Howard Lasnik, Pritty Patel-Grosz, and Charles Yang, editors, *Syntactic Structures after 60 Years*. De Gruyter Mouton.

Noam Chomsky. 2021. Minimalism: Where are we now, and where can we hope to go. *Gengo Kenkyu*, 160:1–41.

Noam Chomsky and George A. Miller. 1963. Introduction to the formal analysis of natural languages. In R. Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology, Volume II*, pages 269–321. Wiley.

Sandra Chung and James McCloskey. 1987. Government, barriers, and small clauses in Modern Irish. *Linguistic Inquiry*, 18(2):173–237.

Guglielmo Cinque. 2017. A microparametric approach to the head-initial/head-final parameter. *Linguistic Analysis*, 41.

Barbara Citko. 2011. Multidominance. In Cedric Boeckx, editor, *Oxford Handbook of Linguistic Minimalism*, pages 119–142. Oxford University Press.

Chris Collins and Richard S. Kayne. 2020. Towards a theory of morphology as syntax. Technical report, New York University.

Robert Daland. 2014. What is computational phonology? *Loquens*, 1(1):e004.

Michael W. Daniels and W. Detmar Meurers. 2004. GIDLP: A grammar format for linearization-based HPSG. In *Procs. 11th Int. Conference on Head-Driven Phrase Structure Grammar*, page 93–111. CSLI.

Hossep Dolatian, Aniello De Santo, and Thomas Graf. 2021. Recursive prosody is not finite-state. *Procs 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, page 11–22.

Nigel Duffield. 1995. *Particles and Projections in Irish Syntax*. Springer.

Jason Eisner. 1997. Efficient generation in primitive optimality theory. In *Procs. 35th Annual Meeting of the Association for Computational Linguistics*.

Emily Elfner. 2012. *Syntax-Prosody Interactions in Irish*. Ph.D. thesis, University of Massachusetts, Amherst.

Mark T. Ellison. 1994. Phonological derivation in optimality theory. In *Procs. 15th Int. Conf. on Computational Linguistics*, pages 1007–1013.

Joost Engelfriet, Eric Lilin, and Andreas Maletti. 2009. Extended multi bottom–up tree transducers: Composition and decomposition. *Acta Informatica*, 46(8):561–590.

Samuel D. Epstein, Hisatsugu Kitahara, and T. Daniel Seely. 2012. Labeling by minimal search. *Linguistic Inquiry*, 45(13):463–481.

Marina Ermolaeva and Gregory M. Kobele. 2021. Agreement as information transmission over dependencies. Technical report, Universität Leipzig. Forthcoming.

Meaghan Fowlie. 2014. Adjunction and minimalist grammars. In *Formal Grammar: 19th International Conference*, pages 34–51. Springer.

Robert Frank and Giorgio Satta. 1998. Optimality theory and the generative complexity of constraint violability. *Computational Linguistics*, 24:307–315.

Hans-Martin Gärtner. 2002. *Generalized Transformations and Beyond*. Akademie Verlag.

Hans-Martin Gärtner. 2014. Strange loops: Phrase-linking grammar meets Kaynean pronominalization. *Linguistische Berichte*, 2014(239):383–395.

Thomas Genet and Valérie Viet Triem Tong. 2001. Reachability analysis of term rewriting systems with Timbuk. In Robert Nieuenhuis and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence and Reasoning*. Springer LNAI 2250. Code: http://www.irisa.fr/lande/genet/timbuk/.

Jean-Yves Girard. 1987. Linear logic. *Theoretical Computer Science*, 50:1–102.

Kyle Gorman and Richard Sproat. 2021. *Finite-State Text Processing*. Morgan & Claypool.

Thomas Graf. 2012a. Concealed reference-set computation: How syntax escapes the parser's clutches. In Anna Maria Di Sciullo, editor, *Towards a Biolinguistic Understanding of Grammar. Essays on Interfaces*, pages 339–362. John Benjamins.

Thomas Graf. 2012b. Reference-set constraints as linear tree transductions via controlled optimality systems. In *Formal Grammar 2010/2011*, pages 97–113. Springer LNCS 7395. Slides.

Thomas Graf. 2018. Why movement comes for free once you have adjunction. In *Procs. Chicago Linguistic Society, CLS 53*, pages 117–136.

Thomas Graf. 2022. Subregular linguistics: Bridging theoretical linguistics and formal grammar. *Theoretical Linguistics*, 48(3-4):145–184.

Thomas Graf and Kalina Kostyszyn. 2021. Multiple wh-movement is not special: The subregular complexity of persistent features in minimalist grammars. In *Procs. Society for Computation in Linguistics*, volume 4, page 26.

Boris Harizanov and Vera Gribanova. 2019. Whither head movement? *Natural Language and Linguistic Theory*, 37:461–522.

Henk Harkema. 2001a. A characterization of minimalist languages. In *Logical Aspects of Computational Linguistics*, LNAI 2099, pages 193–211. Springer.

Henk Harkema. 2001b. *Parsing Minimalist Languages*. Ph.D. thesis, University of California.

Jeffrey Heinz and William J. Idsardi. 2017. Computational phonology today. *Phonology*, 34(2):211–219.

Jeffrey Heinz, Gregory M. Kobele, and Jason Riggle. 2009. Evaluating the complexity of optimality theory. *Linguistic Inquiry*, 40(2):277–288.

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

Mans Hulden. 2009. FOMA: A finite-state compiler and library. In *Procs EACL 2009*.

Tim Hunter. 2011. Insertion minimalist grammars: Eliminating redundancies between merge and move. In *Procs Mathematics of Language 12, LNAI 6878*. Springer.

Tim Hunter. 2015. Deconstructing merge and move to make room for adjunction. *Syntax*, 18(3):266–319.

William J. Idsardi. 2018. Why is phonology different? No recursion. In Ángel J. Gallego and Roger Martin, editors, *Language, Syntax, and the Natural Sciences*, pages 212–223. Cambridge University Press.

Junko Ito and Armin Mester. 2012. Recursive prosodic phrasing in Japanese. In *Prosody Matters: Essays in Honor of Elisabeth Selkirk*, pages 280–303. Elsevier.

Gerhard Jäger. 2002. Gradient constraints in finite state OT: The unidirectional and the bidirectional case. In I. Kaufmann and B. Stiebels, editors, *More than Words: A Festschrift for Dieter Wunderlich*, pages 299–325. Akademie Verlag.

Trevor Jim and Yitzhak Mandelbaum. 2010. Efficient Earley parsing with regular right-hand sides. *Electronic Notes in Theoretical Computer Science*, 253:135–148.

Kyle Johnson. 2017. Rethinking linearization. Technical report, University of Massachussetts, Amherst.

Nicholas Kalivoda. 2018. *Syntax-Prosody Mismatches in Optimality Theory*. Ph.D. thesis, University of California, Santa Cruz.

Nicholas Kalivoda and Jennifer Bellik. 2020. Overtly headed XPs and Irish syntax–prosody mapping. In *Procs. Annual Meetings on Phonology*.

Laura Kallmeyer. 2010. *Parsing Beyond Context-Free Grammars*. Springer.

Makoto Kanazawa, Jens Michaelis, Sylvain Salvati, and Ryo Yoshinaka. 2011. Well-nestedness properly subsumes strict derivational minimalism. In *Logical Aspects of Computational Linguistics*, pages 112–128. Springer LNCS 6736.

Max Kanovich. 2015. Horn linear logic and Minsky machines. *Computing Research Repository*, arXiv:1512.04964.

Lauri Karttunen. 1998. The proper treatment of optimality in computational phonology. In *Procs International Workshop on Finite-State Methods in Natural Language Processing*.

Richard S. Kayne. 1994. *The Antisymmetry of Syntax*. MIT Press.

Richard S. Kayne. 2020. Antisymmetry and externalization. *LingBuzz*, 005554.

S. C. Kleene. 1956. Representation of events in nerve nets and finite automata. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–42. Princeton University Press.

Gregory M. Kobele. 2019. Parsing ellipsis efficiently. In Robert C. Berwick and Edward P. Stabler, editors, *Minimalist Parsing*, pages 110–124. Oxford University Press.

Gregory M. Kobele. 2021. Minimalist grammars and decomposition. Technical report, Universität Leipzig. Forthcoming.

Yusuke Kubota and Robert D. Levine. 2021. *Type-Logical Syntax*. MIT Press.

Leland Kusmer. 2020. *Optimal Linearization: Prosodic Displacement in Khoekhoegowab and Beyond*. Ph.D. thesis, University of Massachusetts, Amherst.

Andrew Lamont. 2022. *Directional Harmonic Serialism*. Ph.D. thesis, University of Massachusetts, Amherst.

Howard Lasnik. 2011. What kind of computing device is the human language faculty? In A. M. di Sciullo and C. Boeckx, editors, *The Biolinguistic Enterprise*. Oxford University Press.

Howard Lasnik and Juan Uriagereka. 2022. *Structure: Concepts, Consequences, Interactions*. MIT Press.

Wim Martens and Joachim Niehren. 2005. Minimizing tree automata for unranked trees. In *Procs 10th Annual Symposium on Database Programming Languages*, pages 232–246. Springer LNCS 3774.

Jonathan May and Kevin Knight. 2006. Tiburon: A weighted tree automata toolkit. In *Proc. 11th International Conference on Implementation and Application of Automata*. Springer LNCS 4904. Code: https://github.com/isi-nlp/tiburon.

James McCloskey. 1999. On the right edge in Irish. *Syntax*, 2:189–209.

James McCloskey. 2002. Resumption, successive cyclicity, and the locality of operations. In Samuel D. Epstein and T. Daniel Seely, editors, *Derivation and explanation in the Minimalist Program*, pages 184–226. Blackwell.

James McCloskey. 2017. New thoughts on old questions – Resumption in Irish. In Jason Ostrove, Ruth Kramer, and Joseph Sabbagh, editors, *Asking the Right Questions: Essays in Honor of Sandra Chung*. University of California, Santa Cruz.

James McCloskey. 2022. The syntax of Irish Gaelic. Technical report, University of Califoria, Santa Cruz. Forthcoming.

Andrew McInnerney. 2022a. Against the argument/adjunct distinction. *Procs. 45th Annual Penn Linguistics Conference*, 28(1).

Andrew McInnerney. 2022b. *The Argument/Adjunct Distinction and the Structure of Prepositional Phrases*. Ph.D. thesis, University of Michigan.

Jens Michaelis. 2001. Transforming linear context free rewriting systems into minimalist grammars. In *Logical Aspects of Computational Linguistics*, pages 228–244. Springer LNCS 2099.

Daniel Milway. 2022. A parallel derivation theory of adjuncts. *Biolinguistics*, 16:e9313.

Richard Moot. 2002. *Proof Nets for Linguistic Analysis*. Ph.D. thesis, Utrecht University.

Richard Moot and Christian Retoré. 2012. *The Logic of Categorial Grammars*. Springer.

Glyn Morrill and Oriol Valentín. 2017. A reply to Kubota and Levine on gapping. *Natural Language and Linguistic Theory*, 35(1):257–270.

Ann E. Mulkern. 2003. *Cognitive Status, Discourse Salience, and Information Structure: Evidence from Irish And Oromo*. Ph.D. thesis, University of Minnesota.

Ann E. Mulkern. 2009. Left right behind: Irish pronoun postposing and information structure. In Andrew Carnie, editor, *Formal Approaches to Celtic Linguistics*. Cambridge Scholars.

Sara Myrberg. 2013. Sisterhood in prosodic branching. *Phonology*, 30(1):73–124.

Ad Neeleman, Joy Philip, Misako Tanaka, and Hans van de Koot. 2023. Subordination and binary branching. *Syntax*, Volume26(1).

Jairo Nunes. 1999. Linearization of chains and phonetic realization of chain links. In Samuel Epstein and Norbert Hornstein, editors, *Working Minimalism*, pages 217–249. MIT Press.

Kenji Oda. 2012. *Issues in the Left Periphery of Modern Irish*. Ph.D. thesis, University of Toronto.

Richard Pattis. 1994. Teaching EBNF first in CS 1. *ACM SIGCSE Bulletin*, 26(1):300–303.

Janet Pierrehumbert and Mary Beckman. 1988. *Japanese Tone Structure*. MIT Press.

Xavier Rival and Jean Goubault-Larrecq. 2001. Experiments with finite tree automata in Coq. In *Theorem Proving in Higher Order Logics*. Springer LNCS 2152. Code: https://github.com/coq-contribs/tree-automata.

Ian G. Roberts. 2021. *Parameter Hierarchies and Universal Grammar*. Oxford University Press.

Sylvain Salvati. 2011. Minimalist grammars in the light of logic. In S. Pogodalla, M. Quatrini, and C. Retoré, editors, *Logic and Grammar*. Springer LNCS 6700.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88:191–229.

Elisabeth Selkirk. 1996. The prosodic structure of function words. In J. Morgan and K. Demuth, editors, *Signal to Syntax*, pages 187–213. Lawrence Erlbaum.

Lisa Selkirk. 2011. The syntax-phonology interface. In *The Handbook of Phonological Theory*, pages 435–484. Wiley-Blackwell.

Stuart M. Shieber. 1984. Direct parsing of ID/LP grammars. *Linguistics and Philosophy*, 7(2):135–154.

Esturo Shima. 2000. A preference for move over merge. *Linguistic Inquiry*, 32(2).

Dominique Sportiche. 2017. Reconstruction, binding, and scope. In M. Everaert and H. van Riemsdijk, editors, *Blackwell Companion to Syntax, 2nd Ed*.

Edward P. Stabler. 1999. Remnant movement and complexity. In G. Bouma, E. Hinrichs, G. Kruijff, and D. Oehrle, editors, *Constraints and Resources in Natural Language Syntax and Semantics*. CSLI.

Edward P. Stabler. 2001. Recognizing head movement. In P. de Groote, G. Morrill, and C. Retoré, editors, *Logical Aspects of Computational Linguistics*, LNAI 2099, pages 254–260. Springer.

Edward P. Stabler. 2011. Computational perspectives on minimalism. In Cedric Boeckx, editor, *Oxford Handbook of Linguistic Minimalism*, pages 617–641. Oxford University Press.

Edward P. Stabler. 2013. Two models of minimalist, incremental syntactic analysis. *Topics in Cognitive Science*, 5(3):611–633.

Miloš Stanojević. 2019. On the computational complexity of head movement and affix hopping. In *Formal Grammar, 24th International Conference*, pages 101–116. Springer LNCS 11668.

Miloš Stanojević and Mark Steedman. 2021. Formal basis of a language universal. *Computational Linguistics*, 47(1):9–42.

John Torr and Edward P. Stabler. 2016. Coordination in minimalist grammars. In *Procs. 12th Workshop on Tree-Adjoining Grammars and Related Formalisms*.

Mai Ha Vu, Nazila Shafiei, and Thomas Graf. 2019. Case assignment in TSL syntax. *Procs. Society for Computation in Linguistics*, pages 267–276.

Philip Wadler. 1992. Comprehending monads. *Mathematical Structures in Computer Science*, 2(4):461–493.

Michael Wagner. 2010. Prosody and recursion in coordinate structures and beyond. *Natural Language and Linguistic Theory*, 28(1):183–237.

Joseph W. Windsor, Stephanie Coward, and Darin Flynn. 2018. Disentangling stress and pitch accent in Munster Irish. In *Procs 35th West Coast Conference on Formal Linguistics*. Cascadilla.

Niklaus Wirth. 1977. What can we do about the unnecessary diversity of notation for syntactic definitions? *Communications of the ACM*, 20(11):822–823.

Kristine M. Yu. 2021. Computational perspectives on phonological constituency and recursion. *Catalan Journal of Linguistics*, 77:114.

Kristine M. Yu. 2022. Representations for multiple dependencies in prosodic structures. *Proceedings of the Society for Computation in Linguistics*, 5:15.

A Weak equivalence of $*$-extensions: Sketch

Since $*$-MG extends MG, it is trivially true that $L(* - MG) \supseteq L(MG)$, and similarly for $*$-L(MCFG). Since $L(MG) = L(MCFG)$ (Harkema, 2001a; Michaelis, 2001), $L(*-MG) \subseteq L(MCFG)$ can be established by showing $L(* - MG) \subseteq L(MCFG)$. When labeling allows unbounded branching in the MG, a corresponding $*$-MCFG rule can be formulated. To construct a

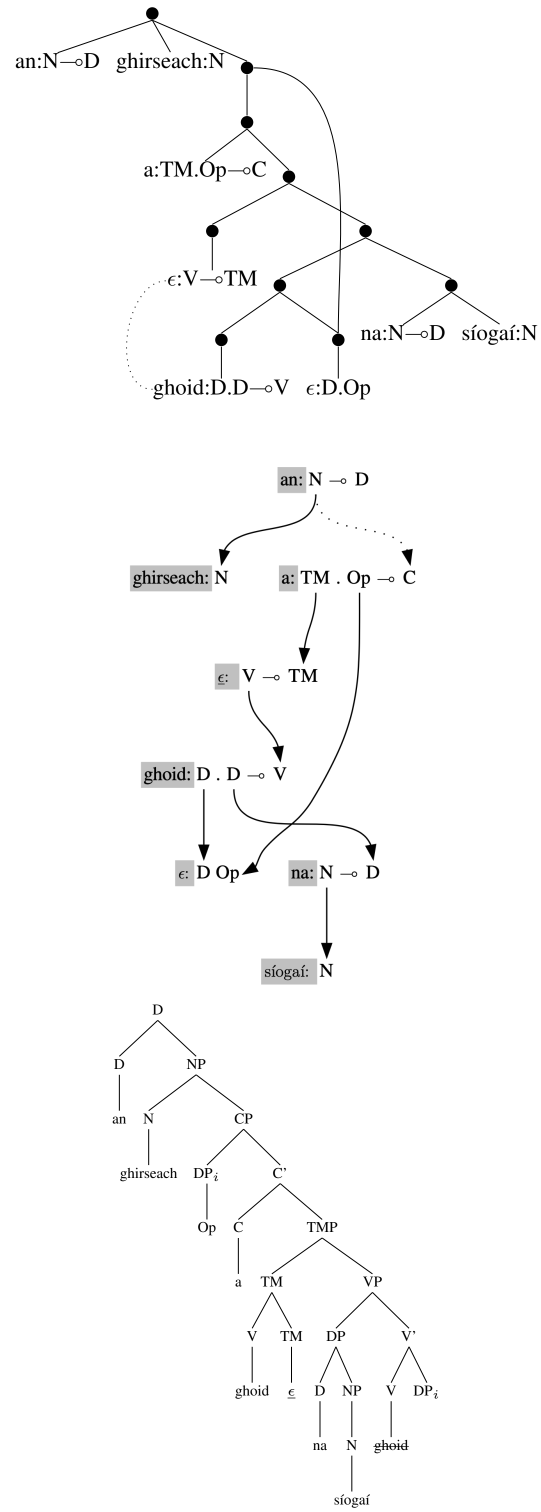


Figure 6: A set (leaves are lexical items, internal nodes are sets, arcs are $\in$ relations, with a dotted arc for head movement), dependency tree (with solid feature-checking arcs and dotted adjunct arc), and X-bar tree (for linguists) for (5a).

weakly equivalent MCFG, we simply replace unbounded branching with corresponding right recursive rules and prove the language is unchanged.

B Adjuncts and wh-movement

The approach used for coordination in the text is easily adapted to McInnerney (2022b)'s proposal, mentioned in the introduction, that unboundedly many adjuncts can be merged as sisters of the head they modify. His analysis is motivated in large part by a labeling theory that aims to reduce stipulated features, but as a place-holder for that kind of revision, here we simply extend our feature-based labeling to adjuncts.[15] It suffices to extend the definition of $\&(\gamma, \vec{C})$ in Figure 2 with one that is true whenever each element of C has a label of an admissible adjunct of γ.

In some dialects of Irish, when there is an Ā-extraction, as in the relative clause of (5a) from McCloskey (2002, (9)), the complementizer is pronounced differently than when there is resumption instead of extraction, as in (5b):[16]

(5) a. an ghirseach a ghoid na síogaí
 the girl aL stole the fairies
 'the girl that the fairies stole away'

 b. an ghirseach a-r ghoid na síogaí í
 the girl aN-[PAST] stole the fairies her
 'the girl that the fairies stole away'

As a step towards MG implementation, let the relevant EPP/operator feature of aN be Op, in a relative clause adjoined as sister to the head N, in the structure for (5a) of Figure 6. Any number of additional adjuncts could occur as sister to the noun and relative clause.

C Implementation

Implementations of nondeterminism can be easy in programming languages like SWI Prolog that provide backtracking search. Represent {A,B} with the term `[A,B]` and $phon : a_1 \ldots a_i \multimap a_{i+1} \ldots a_{i+j}$ with $\texttt{[phon]-[a}_1\texttt{,...,a}_i\texttt{]-[a}_{i+1}\texttt{,...,a}_{i+j}\texttt{]}$. Then this 10 clause prolog implementation of R

[15] See McInnerney (2022b,a) on binding phenomena and other considerations that motivated the more common hierarchical analyses of adjunction. Cf. also Milway (2022); Graf (2018); Hunter (2015, 2011); Fowlie (2014).

[16] See McCloskey (2002, 2017); Oda (2012) and references cited there for careful discussion. Agreement and other relevant considerations are beyond the scope of this brief paper; see e.g. Ermolaeva and Kobele (2021) on agreement in an MG-based framework, Vu et al. (2019) on case.

calls itself recursively until complete structure A is generated from numeration X0, if possible:[17]

```
r([A], A) :- l(A, []-[_]-[]).
r(X0, X) :- select(A, X0, X1), l(A, [F0|AN]-AP-AC0),
    ( nonvar(F0), F0=p(F) -> P=true ; F0=F, P=fail),
    ( select([F|BP]-B, AC0, AC) -> X2 = X1, BC = []
    ; select(B, X1, X2), l(B, []-[F|BP]-BC), AC=AC0
    ), m(B, [F|AN]-AP-AC, []-[F|BP]-BC, ABF),
    '&'(P, F, X2, X3, Cs), mrg(A, B, Cs, ABCs),
    r([ABCs|X3], X).
mrg(A, B, Cs, [A,B|Cs]).
l(_-A-B, A-B-[]).
l([A,B], D) :- l(A, AF), l(B, BF), m(B, AF, BF, D).
m(B, [F0|AN]-AP-AC, BF, AN-AP-ABC) :-
 (nonvar(F0), F0=p(F) -> true ; F0=F),
 (select([F|BP]-B0, AC, AC1) -> B0=B,
   (BP = [] -> AC1 = ABC ;
    BP = [G|BP1], smc([[G|BP1]-B0|AC1], [], ABC, []))
 ; BF = []-[F]-BC, smc(AC, BC, ABC, [])
 ; BF = []-[F,G|BP]-BC, smc([[G|BP]-B|AC], BC, ABC, [])).
'&'(_,_,X,X,[]).
'&'(true,F,X0,X,[C|Cs]) :-
 select(C,X0,X1), l(C,[]-[F]-[]), '&'(true,F,X1,X,Cs).
smc([], D, D, _).
smc([[F|C]-A|L], M, [[F|C]-A|N], Fs) :-
    \+member(F, Fs), smc(L, M, N, [F|Fs]).
```

Derived structures here are lists not sets, but order of elements is irrelevant except for for identification of the head, and that is always determined by features alone. All syntactic structures in the text can be computed by this implementation. For example, this session computes the structure shown in Figure 3:

```
?- r([[]-[tm]-[c],[]-[v]-[tm],[is]-[a]-[v],
      [cuma]-[lpred]-[a],[]-[pred,d]-[lpred],
      [na]-[d]-[pred],[shamhradh]-[]-[d],
      [na]-[d]-[pred],[fhomar]-[]-[d],
      [na]-[d]-[pred],[gheimhreadh]-[]-[d],
      [no]-[X,p(X)]-[X],[e]-[]-[d]],A).

A = [
      []-[tm]-[c],
      [
        []-[v]-[tm],
        [
          [is]-[a]-[v],
          [
            [cuma]-[lpred]-[a],
            [
              [
                []-[pred,d]-[lpred],
                [
                  [
                    [no]-[pred,p(pred)]-[pred],
                    [
                      [na]-[d]-[pred],
                      [gheimhreadh]-[]-[d] ] ],
                [
                  [na]-[d]-[pred],
                  [fhomar]-[]-[d] ],
                [
                  [na]-[d]-[pred],
                  [shamhradh]-[]-[d] ] ] ],
          [e]-[]-[d] ] ] ] ] ].
```

As discussed in §4, efficiently implementing rule K, for parsing, requires more bookkeeping. In the deductive format of Stabler (2011, §A), for rule K, the feature checking rules for external merge (EM) with f^+ where $\sqcup$ is smc-respecting union:

$$\frac{s{::}f^+\alpha{-\!\!\circ}\beta,\gamma_1 \quad t{\cdot}f,\gamma_2}{st{:}\alpha{-\!\!\circ}\beta,\gamma_1{\sqcup}\gamma_2} \text{ (EM1}^+\text{)}$$

$$\frac{s{::}f^+\alpha{-\!\!\circ}\beta,\gamma_1 \quad t{\cdot}f,\gamma_2}{st{:}f^+\alpha{-\!\!\circ}\beta,\gamma_1{\sqcup}\gamma_2} \text{ (EM1}^{++}\text{) if } t \neq \epsilon$$

$$\frac{s{:}f^+\alpha{-\!\!\circ}\beta,\gamma_1 \quad t{\cdot}f,\gamma_2}{ts{:}\alpha{-\!\!\circ}\beta,\gamma_1{\sqcup}\gamma_2} \text{ (EM2}^+\text{)}$$

$$\frac{s{:}f^+\alpha{-\!\!\circ}\beta,\gamma_1 \quad t{\cdot}f,\gamma_2}{ts{:}f^+\alpha{-\!\!\circ}\beta,\gamma_1{\sqcup}\gamma_2} \text{ (EM2}^{++}\text{) if } t \neq \epsilon$$

$$\frac{s{\cdot}f^+\alpha{-\!\!\circ}\beta,\gamma_1 \quad t{\cdot}f\delta,\gamma_2}{s{:}\alpha{-\!\!\circ}\beta,\gamma_1{\sqcup}\gamma_2{\sqcup}\{t{:}\delta\}} \text{ (EM3}^+\text{) if } \delta \neq \epsilon.$$

Note that the Kleene + introduces indeterminacy, reflected here by the two rules for each of EM1 and EM2. The second case for external merge of a 'mover', EM3, and movement rules for these cases require a treatment of ATB movement – left for future work. The rules for f^* are the same, except that f^* is also 'checked' by 0 positive occurrences. See link in fn 17 for a working implementation.

The extension to rules for head movement can follow Stabler (2001); Stanojević (2019). The examples in the paper and the rules shown here only consider negative occurrences of f^+ and f^*. Positive occurrences may subsume previous proposals about 'persistent features', relevant for successive cyclic movement – left for future work.

The first steps toward a GEN transduction for prosody, discussed in §3, are also easily implemented. Represent a tree with root A and daughters B,C,D by the prolog term `A/[B,C,D]`. Then a relation that pairs the X-bar structure in Figure 4 – without coindexing and without the second coordinate – with the prosodic structure in Figure 1b, is computed by the following implementation:

```
head(X) :- member(X, [c,tm,v,a,lpred,pred,b]).
phrase(XP) :- atom_chars(XP, L), last(L, p).
gen(T, Out) :- rule(T, Out).
gen(X/L,T) :-  maplist(gen,L,S), rule(X/S,T).
rule(_/[qw/[X0], qphi/[X1]], qi/[i/[X0, X1]]).
rule(X/[Ph/[]], qw/[w/[Ph/[]]]) :- head(X).
rule(X/[Ph/[]], qphi/[phi/[w/[Ph/[]]]]) :- phrase(X).
rule(_/[], qe).
rule(_/[qw/[X0], qphi/[X1]], qphi/[phi/[X0,X1]]).
rule(_/[qphi/[X0], qphi/[X1]], qphi/[phi/[X0,X1]]).
rule(_/[qe, X0], X0).
rule(_/[X0, qe], X0).
rule(_/[X0], X0).
```

Representing the reduced Figure 4 by a prolog term, as the first argument to `gen`, this code computes the prolog term for Figure 1b as the first of many candidate structures.

[17] This code (with some explanatory comments!) is available at https://github.com/epstabler/star, along with display tools, a parser for rule K (in python), and tree transducers.