Tupled Pregroup Grammars

Edward P. Stabler
University of California, Los Angeles

Abstract. This paper extends Lambek's pregroup grammars in a way designed to facilitate comparison with proposals in the tradition of Chomskian syntax. Instead of categorizing expressions with sequences of pregroup terms, we use tuples of sequences of terms and define basic operations on tuples. Then we can define non-context free languages and realize some recent syntactic analyses quite straightforwardly. The pregroup operations provide a simple 'feature checking' and the tupling allows operations rather like 'movement'. A certain kind of tupled pregroup grammar (mTPG) is 'mildly context sensitive', in Joshi's sense, and seems to allow quite natural analyses for 'head movement', 'pied piping' and 'remnant movement' constructions, avoiding special, ad hoc mechanisms.

Constituent combination is treated in classical categorial grammar as kind of non-associative product, and there is no corresponding division operation (Bar-Hillel, 1953). Lambek noticed that associativity and division are easily added (Lambek, 1958), but the resulting calculus is rather complex: even the proof that it defines exactly the (ϵ-free) context free languages is non-trivial (Pentus, 1993). Lambek, Casadio, Buszkowski and others have more recently studied simpler associative calculi they call 'pregroup grammars', and an easy proof shows that they also define the context free languages.

This paper extends pregroup grammars by making them higher order, in a way designed to facilitate comparison with proposals in the tradition of Chomskian syntax. Categorizing expressions with k-tuples (or equivalently, k-ary functions) of sequences of terms instead of merely sequences of terms, and defining basic operations on tuples, non-context free languages can be defined. In fact, a certain kind of tupled pregroup grammar (mTPG) presented here is 'mildly context sensitive' in the sense of Joshi (1985), with exactly the same expressive power as the set-local multi-component tree adjoining grammars of Weir (1988), the multiple context free grammars (MCFGs) of Seki et al. (1991) and the minimalist grammars (MGs) of Stabler and Keenan (2003). mTPG operations are much simpler than MG operations, and furthermore, whereas some 'head movement', 'affix hopping', 'partial movement' and 'pied piping' phenomena seem to motivate complex, ad hoc mechanisms in minimalist grammars (Stabler, 2001), mTPGs appear to provide reasonable treatments without any extension.

DRAFT COMMENTS WELCOME: stabler@ucla.edu

1. Pregroup grammars and blocking constraints

Following Lambek (2001) and Buszkowski (2001), consider any set $\mathbb{P}$ of **simple types**, and define an associated set of **terms** $\mathbb{R}_{\mathbb{P}}$ as the closure of $\mathbb{P}$ with respect to function l together with the closure of $\mathbb{P}$ with respect to function r. These 'left adjoint' and 'right adjoint' functions are usually written as superscripts, so for any simple type $\mathrm{a} \in \mathbb{P}$, $\mathbb{R}_{\mathbb{P}}$ contains the following, and we will assume they are pairwise distinct:

$$\ldots \mathrm{a}^{\mathrm{ll}} \quad \mathrm{a}^{\mathrm{l}} \quad \mathrm{a} \quad \mathrm{a}^{\mathrm{r}} \quad \mathrm{a}^{\mathrm{rr}} \ldots$$

It is convenient to introduce another notation for terms too, representing any simple type $\mathrm{a} = \mathrm{a}^{(0)}$, iterations of the right adjoint with positive superscripts in parentheses, and iterations of the left adjoint with negative superscripts in parentheses, so the previous sequence is:

$$\ldots \mathrm{a}^{(-2)} \quad \mathrm{a}^{(-1)} \quad \mathrm{a}^{(0)} \quad \mathrm{a}^{(1)} \quad \mathrm{a}^{(2)} \ldots$$

As usual, for any set S, S^* is the set of finite sequences of elements of S, and ϵ is the empty sequence. Define the set of **types** $\mathbb{T}_{\mathbb{P}} = \mathbb{R}^*_{\mathbb{P}}$, the finite sequences of terms. (We will often drop the subscripts from $\mathbb{T}_{\mathbb{P}}$ and $\mathbb{R}_{\mathbb{P}}$ when no confusion will result.) Over alphabet Σ and simple types $\mathbb{P}$, **PG expressions** are type, string pairs, elements of $\mathbb{E} = \mathbb{T}_{\mathbb{P}} \times (\Sigma \cup \{\epsilon\})^*$, sometimes written t:s, and sometimes with types over strings $\binom{t}{s}$.

A **pregroup grammar** $\mathrm{G} = \langle \Sigma, \mathbb{P}, \leq, \mathbb{I}, \mathrm{S} \rangle$ where Σ is a nonempty alphabet, $\mathbb{P}$ is a set of simple types partially ordered by $\leq$, finite set $\mathbb{I} \subset \mathbb{T}_{\mathbb{P}} \times ((\Sigma \cup \{\epsilon\})$, and 'start' type $\mathrm{S} \in \mathbb{P}$.

Let $\twoheadrightarrow_{PG}$ be the binary relation on $\mathbb{E}^*$ that holds only in the following three cases, for any $\alpha, \beta \in \mathbb{E}^*$ and any $a, b \in \mathbb{T}$ such that either $a \leq b$ and n is even, or $b \leq a$ and n is odd,

$$\text{(Conc)} \qquad \alpha \binom{t_1}{s_1} \binom{t_2}{s_2} \beta \twoheadrightarrow_{PG} \alpha \binom{t_1 t_2}{s_1 s_2} \beta.$$

$$\text{(GCON)} \qquad \alpha \binom{x a^{(n)} b^{(n+1)} y}{s} \beta \twoheadrightarrow_{PG} \alpha \binom{xy}{s} \beta$$

$$\text{(GEXP)} \qquad \alpha \binom{xy}{s} \beta \twoheadrightarrow_{PG} \alpha \binom{x a^{(n+1)} b^{(n)} y}{s} \beta.$$

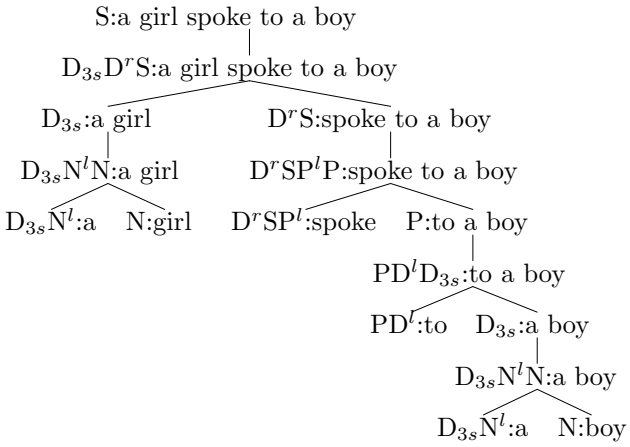
Define the powers $s \twoheadrightarrow^0_{PG} t$ iff $s = t$, and $s \twoheadrightarrow^{k+1}_{PG} t$ iff $\exists u,\ s \twoheadrightarrow^k_{PG} u$ and $u \twoheadrightarrow_{PG} t$. Let $s \twoheadrightarrow^*_{PG} t$ iff $\exists k,\ s \twoheadrightarrow^k_{PG} u$. Then the language $\mathrm{L_G} = \{s |\ \mathbb{I}^* \twoheadrightarrow^*_{PG} \binom{\mathrm{S}}{s}\}$.

1.1. Example G_0.

Let the vocabulary $\Sigma = \{\text{a, boy, girl, laughs, praised, and, spoke, to}\}$, the simple types $\mathbb{P} = \{D_{3s}, D, N, S\}$ with $\leq$ the identity together with $D_{3s} \leq D$, and with the following 8 expressions in the lexicon $\mathbb{I}$:

$$\binom{D_{3s}N^l}{\text{a}}\binom{N}{\text{boy}}\binom{N}{\text{girl}}\binom{D^rDD^l}{\text{and}}\binom{D^r_{3s}S}{\text{laughs}}\binom{D^rSD^l}{\text{praised}}\binom{D^rSP^l}{\text{spoke}}\binom{PD^l}{\text{to}}.$$

Recall the notational convention that $ab^r = a^{(0)}b^{(1)}$, and since 0 is even, this contracts if $a \leq b$, by GCON above. But $a^lb = a^{(-1)}b^{(0)}$, and so contracts if $b \leq a$. So if we read $\leq$ as 'is an instance of', then GCON says that contraction happens when the adjoint of a term is appropriately adjacent to an instance of that term. So G_0 generates an infinite language containing *a boy laughs* but not *a boy and a girl laughs*, as desired. We can depict derivations in a standard tree format:



Binary branching indicates applications of Conc, unary branching indicates applications of GCON, and each leaf is an element of the lexicon $\mathbb{I}$. For any G, to derive $\binom{t}{s} \in L(G)$ with $t \in \mathbb{P}$, GEXP is never required (Lambek, 1999, Prop.2), recalling the 'cut elimination' result for the calculus of Lambek (1958). The reader can easily verify that, even without GEXP, there are many derivations of this string. With a computer implementation it is easy to calculate that there are 12,016 different derivation trees which could have been displayed here.[1]

[1] With Lemma 9 below, parsing methods for MGs are easily adapted to the systems defined here (PGs, mPGs, mTPGs). See implementations at `http://www.linguistics.ucla.edu/people/stabler/`.

1.2. A BLOCKING CONSTRAINT

We follow Lambek (2000) in proposing that some undesirable derivations should be blocked with 'performance constraints', but pursue a different idea about what the constraints are, one inspired by the tradition of Chomskian syntax. Notice that the type of each lexical item in G_0 has exactly one one element of $\mathbb{P}$, and in the derivation shown, no Conc step applies to a type with more than one element of $\mathbb{P}$ in it. Derivations like this are common in linguistic applications. Let's say a type **proper** iff GCON does not apply to it and it has exactly 1 element of $\mathbb{P}$; it is **saturated** iff no adjoint types occur in it; and it is **rl** iff it has 0 or more right adjoints, followed by a simple type, followed by 0 or more left adjoints. (So an rl type is proper, and has no adjoints of adjoints.) An expression is **proper (saturated, rl)** iff its type is. A **minimal PG (mPG)** is a PG in which

$\mathbb{I}$ assigns rl types;

Conc applies to a pair of expressions only if **both are proper** and **at least one of them is saturated**.[2]

Some restriction of this sort might be motivated by a memory limitation – an inability to remember more than one incomplete, unsaturated element at at time (cf. Stabler 1994).

Above we showed 1 of 12,016 derivation trees for the sentence *a girl spoke to a boy*, but only in 2 of these does Conc respect the proper and saturation requirements, the one above with the constituency [*a girl*][*spoke* [*to* [*a boy*]]], and another with the constituency [[*a girl*] *spoke*][*to* [*a boy*]]. The way proper typing and saturation induce structure here is key. There is only one analysis of the P phrase *to a boy* because *to* and *a* are both unsaturated; but there are two analyses of the major constituents of the sentence, DVP, because both D and P are saturated in this simple grammar. So mPG restrictions remove a good deal of the associativity in the simple system; intuitively, mPG is more associative than the classical categorial grammar, but lacks the global associativity of PG.[3]

So the mPG requirements are severe, but they have no impact on expressive power: the PG-definable and mPG-definable languages are the same, as shown in §2. PGs can be made more expressive with

[2] This is analogous to the assumption in Chomskian syntax that a head projects to the phrasal level – combining with its arguments to the left and right – before it can merge with another phrase or enter into a movement relation. Grammars with movement allow a fully projected phrase to be incomplete in another way too: with 'licensing' requirements, which will find an analog in TPGs introduced below.

[3] Cf. Bar-Hillel (1953), and the modern perspective of Moortgat (1996, ex.2.21).

the addition of modal operators (Fadda, 2002; Oehrle, 2002) but it is not yet clear how to tailor this approach to define exactly the class of languages we want for linguistic theory. In §3, we introduce tupled pregroup grammars (TPGs), and extend our 'performance' restriction to define a well-known mildly context sensitive class of languages.

2. The power of mPGs

First, observe that there are proper lexicons which allow PG derivations of a result even when there are no mPG derivations of the same result. Consider for example this sequence of 3 proper elements:

$$\begin{pmatrix} \mathrm{S} \\ \epsilon \end{pmatrix} \begin{pmatrix} \mathrm{PQ}^{\mathrm{l}} \\ \mathrm{x} \end{pmatrix} \begin{pmatrix} \mathrm{QP}^{\mathrm{r}} \\ \mathrm{y} \end{pmatrix} \twoheadrightarrow_{\mathrm{PG}} \begin{pmatrix} \mathrm{S} \\ \mathrm{xy} \end{pmatrix}.$$

And there are PG derivations of an rl result from rl premises when there are no mPG derivations:

$$\begin{pmatrix} \mathrm{A}^{\mathrm{r}}\mathrm{BC}^{\mathrm{l}} \\ \mathrm{x} \end{pmatrix} \begin{pmatrix} \mathrm{CD}^{\mathrm{l}} \\ \mathrm{y} \end{pmatrix} \twoheadrightarrow^{*}_{\mathrm{PG}} \begin{pmatrix} \mathrm{A}^{\mathrm{r}}\mathrm{BD}^{\mathrm{l}} \\ \mathrm{xy} \end{pmatrix}.$$

Nevertheless we have the following basic results.

LEMMA 1. *Every rl type is proper. And a rl type is saturated iff it is atomic.*

Proof: This follows immediately from the definitions. □

LEMMA 2. *If* $e_1 \dots e_n \twoheadrightarrow^{*}_{\mathrm{PG}} a$ *for rl expressions* $e_1, \dots, e_n$ *(for* $n \geq 1$*) and atomic expression* a*, then there is at least one atomic* e_i $(1 \leq i \leq n)$

Proof: In an unsaturated expression there are at least as many adjoints as atoms, but if $e_1 \dots e_n \twoheadrightarrow^{*}_{\mathrm{PG}} a$ there must be exactly 1 more atom than adjoints in the premises. □

LEMMA 3. *If* $e_1 \dots e_n \twoheadrightarrow^{*}_{\mathrm{PG}} a$ *for rl expressions* $e_1, \dots, e_n$ *(for* $n \geq 1$*) and atomic expression* a*, every GEXP-free PG derivation of this result has exactly* $n-1$ *Conc steps and* $n-1$ *GCON steps.*

Proof: Since Conc is the only way to merge two elements, there must be $n-1$ Conc steps to produce a single expression. Furthermore, since each of the n premises contains exactly 1 element of $\mathbb{P}$, in a GEXP-free derivation, GCON must apply $n-1$ times to yield an atomic expression. □

LEMMA 4. *For any rl expressions e_1, e_2, if GCON(Conc(e_1e_2)) exists, it is rl.*

Proof: Consider any rl expressions e_1, e_2 with the types

$$t_1 = a^l_1 \dots a^l_h b c^r_1 \dots c^r_i \qquad t_2 = d^l_1 \dots a^l_j e f^r_1 \dots f^r_k,$$

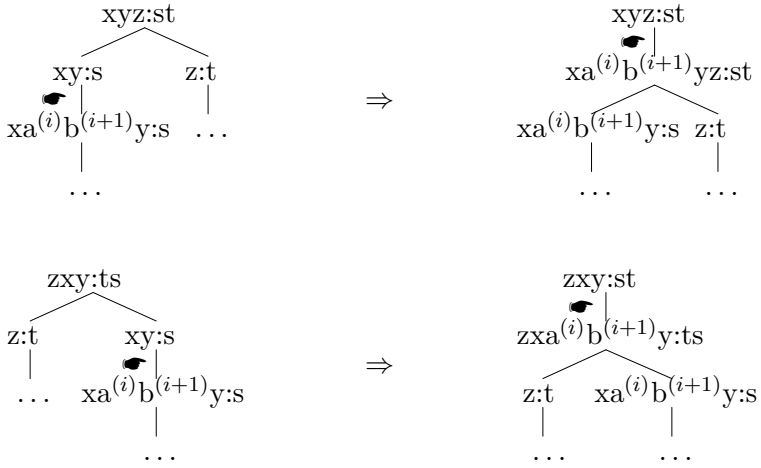
respectively, for some $h, i, j, k \geq 0$ such that GCON can apply to $\mathrm{Conc}(e_1, e_2)$ which has the type

$$a^l_1 \dots a^l_h b c_1 \dots c^r_i d^l_1 \dots a^l_j e f_1 \dots f^r_k.$$

Since GCON applies, $i + j = 1$ so that GCON can delete either b or e, depending on which of i or j is 0, and the result is rl again. □

LEMMA 5. *If $e_1 \dots e_n \twoheadrightarrow^*_{\mathrm{PG}} e$, then it has a derivation in which all Conc steps precede all GCON steps.*

Proof: Consider a GCON step that cancels some $\mathrm{a}^{(i)}\mathrm{b}^{(i+1)}$ immediately before a Conc step, anywhere in any GEXP-free PG derivation. It is clear from the definitions of GCON and Conc that we can move the GCON step up the left or right branch to apply after the Conc step, deriving the same result:



We can apply this step a finite number of times to change any derivation into one in which all Conc steps are at the bottom of the derivation tree, with a sequence of GCON steps following them at the top. □

Let's say that the **size of a derivation** is the number of terms that appear in it. Whenever we move a GCON upward across a Conc step, we increase the size of the derivation by 2. Also, we can observe

LEMMA 6. *We can move a GCON contraction of $a^{(i)}b^{(i+1)}$ "down" in a derivation tree across either a Conc or GCON step, so long as the contracted pair $a^{(i)}b^{(i+1)}$ is present (and adjacent) at the earlier step.*

LEMMA 7. *For rl expressions $e_1, \ldots, e_n$ and atomic a, $e_1 \ldots e_n \twoheadrightarrow^*_{\mathrm{PG}} a$ iff $e_1 \ldots e_n \twoheadrightarrow^*_{\mathrm{mPG}} a$.*

Proof: The ($\Leftarrow$) direction is trivial, so we need only show ($\Rightarrow$). Assuming $e_1 \ldots e_n \twoheadrightarrow^*_{\mathrm{PG}} a$ where the premises are rl and the conclusion atomic, we show $e_1 \ldots e_n \twoheadrightarrow^*_{\mathrm{mPG}} a$ by induction on n. If $n = 1$, then $e_1 = a$ and the result is immediate. So suppose that (IH) for $m \leq n$, $e_1 \ldots e_m \twoheadrightarrow^*_{\mathrm{PG}} a$ iff $e_1 \ldots e_m \twoheadrightarrow^*_{\mathrm{mPG}} a$, with rl premises and atomic conclusion. Now assume $e_1 \ldots e_{m+1} \twoheadrightarrow^*_{\mathrm{PG}} a$. By Lemma 2, we know that there is an an atomic e_i ($1 \leq i \leq m+1$), and that GCON applies to either $\mathrm{CONC}(e_{i-1}e_i)$ or to $\mathrm{CONC}(e_ie_{i+1})$. Suppose the former case (the latter case can be handled similarly). By Lemma 5, $e_1 \ldots e_{m+1} \twoheadrightarrow^*_{\mathrm{PG}} a$ has a derivation in which all Conc steps precede all GCON steps, and furthermore, since Conc is associative, we can apply the initial Conc steps in any order. So consider any Conc-steps-first derivation in which Conc applies to e_{i-1} and e_i. After this step, GCON can apply, and so by Lemma 6, we can move the cancellation of e_i with its adjoint "down" in the tree so that it immediately follows this Conc step. By Lemma 4, we know that the result $\mathrm{GCON}(\mathrm{Conc}(e_{i-1}e_i))$ is rl, and so we know

$$e_1 \ldots e_{i-2}\mathrm{GCON}(\mathrm{Conc}(e_{i-1}e_i))e_{i+1} \ldots e_{m+1} \twoheadrightarrow^*_{\mathrm{PG}} a.$$

Since by the IH, this result has an mPG derivation, we see that

$$e_1 \ldots e_{m+1} \twoheadrightarrow^*_{\mathrm{PG}} a$$

does too. □

THEOREM 1. *Among the PG derivations of any sentence from an rl grammar, the mPG derivations are minimal.*

Proof sketch: We observed above that the size of a derivation is reduced by keeping GCON steps as low as possible, and we see by the construction in the previous proof that mPG derivations always apply Conc to produce a result to which GCON applies immediately, minimizing the size of the derivation. □

Lemma 7 relates PG and mPG derivations with rl premises, but with results from the prior literature, it is now easy to relate all PG languages (whether defined with rl lexicons or not) to mPG languages. First, let a **classical categorial grammar (CG)** $\mathrm{G} = \langle \Sigma, \mathbb{P}, \mathbb{I}, \mathrm{S} \rangle$ where

Σ is a nonempty alphabet, $\mathbb{P}$ is a set of simple types, finite set $\mathbb{I} \subset$ closure$(\mathbb{P}, \{\backslash, /\}) \times ((\Sigma \cup \{\epsilon\})$, and 'start' type $\mathrm{S} \in \mathbb{P}$. Let $\rightarrow^*_{CG}$ be the reflexive, transitive closure of the binary relation defined by the following two axioms:

$$\alpha \begin{pmatrix} A/B \\ s_1 \end{pmatrix} \begin{pmatrix} B \\ s_2 \end{pmatrix} \beta \rightarrow_{CG} \alpha \begin{pmatrix} A \\ s_1 s_2 \end{pmatrix} \beta. \qquad \alpha \begin{pmatrix} B \\ s_1 \end{pmatrix} \begin{pmatrix} B\backslash A \\ s_2 \end{pmatrix} \beta \rightarrow_{CG} \alpha \begin{pmatrix} A \\ s_1 s_2 \end{pmatrix} \beta.$$

Such a grammar defines the **language** $\mathrm{L_G} = \{s |\ \mathbb{I}^* \rightarrow^*_{CG} \begin{pmatrix} \mathrm{S} \\ s \end{pmatrix}\}$.

THEOREM 2. *Language L is PG-definable iff it is mPG-definable.*

Proof: Lemma 7 establishes that all mPG languages are PG languages, so we need only show that all PG languages are mPG languages. Buszkowski (2001) shows that the PG languages are exactly the CG languages, which can be generated by CGs in which the types are all of the form $p, p/q, (p/q)/r$, for atoms p, q, r.[4] So it suffices to show that any language defined by such a CG language is an mPG language. Given any CG $\mathrm{G} = \langle \Sigma, \mathbb{P}, \mathbb{I}, \mathrm{S} \rangle$, it is easy to define an mPG $\mathrm{G}' = \langle \Sigma, \mathbb{P}, \leq, \mathbb{I}', \mathrm{S} \rangle$ that generates the same language. We let $\leq$ be the identity on $\mathbb{P}$ and let $\mathbb{I}'$ be the result of converting the CG types of $\mathbb{I}$ into PG types with the following mapping:

$$p \mapsto p \qquad p/q \mapsto pq^l \qquad (p/q)/r \mapsto pq^l r^l.$$

The equivalence is easily shown by an induction on derivation length: each CG step corresponds to a Conc plus GCON sequence. □

3. Tupled pregroup grammars

PG expressions are typed strings, combined by concatenation, but **TPG expressions** are tuples of typed strings, elements of $\mathbb{E} = (\mathbb{T}_{\mathbb{P}} \times (\Sigma \cup \{\epsilon\})^*)^*$, combined in two steps: a kind of tuple-union called 'merge', and concatenation of arbitrary coordinates of a tuple called 'move'. Expressions will sometimes be written with the types over the strings:

$$\begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix}.$$

Alternatively, we sometimes write the types separated by commas, followed by a colon, followed by the strings separated by commas:

$$t_1, \ldots, t_k : s_1, \ldots, s_k.$$

[4] Buszkowski excludes type assignments to ϵ, which is natural in semigroup theory but unecessary in the present context.

We define a merge operation which applies to any pair of tuples:

$$\begin{pmatrix} t_1 & \dots & t_i \\ s_1 & \dots & s_i \end{pmatrix} \bullet \begin{pmatrix} t_{i+1} & \dots & t_k \\ s_{i+1} & \dots & s_k \end{pmatrix} = \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}.$$

And for any k-tuple ($k > 0$) and any $1 \leq i \leq k$, we define an operation that deletes the i'th coordinate:

$$\begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}_{-i} = \begin{pmatrix} t_1 & \dots & t_{i-1} & t_{i+1} & \dots & t_k \\ s_1 & \dots & s_{i-1} & s_{i+1} & \dots & s_k \end{pmatrix}.$$

A **tupled pregroup grammar (TPG)** $\mathrm{G} = \langle \Sigma, \mathbb{P}, \leq, \mathbb{I}, \mathrm{S} \rangle$ exactly like a PG, except now $\mathbb{I} \subset (\mathbb{T}_{\mathbb{P}} \times (\Sigma \cup \{\epsilon\}))^*$.

Let $\twoheadrightarrow_{TPG}$ be the binary relation on $\mathbb{E}^*$ that holds only in the following 4 cases. For any tuples e_1, e_2 and sequences of tuples α, β:

$$\text{(Mrg)} \qquad \alpha\ e_1\ e_2\ \beta \twoheadrightarrow_{TPG} \alpha\ e_1 \bullet e_2\ \beta.$$

(Move) applies to any k-tuple ($k > 1$), for any $1 \leq i \leq k$ and $1 \leq j < k$,

$$\text{(Move)} \qquad \alpha \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix} \beta \twoheadrightarrow_{\mathrm{TPG}} \alpha \begin{pmatrix} t_i t_j \\ s_i s_j \end{pmatrix} \bullet \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}_{-\mathrm{i}-\mathrm{j}} \beta.$$

The type in any coordinate can be contracted or expanded, for any $a, b \in \mathbb{T}$ such that either $a \leq b$ and n is even, or $b \leq a$ and n is odd:

$$\text{(GCON)} \qquad \alpha \begin{pmatrix} \dots & x a^{(n)} b^{(n+1)} y & \dots \\ & s & \end{pmatrix} \beta \twoheadrightarrow_{TPG} \alpha \begin{pmatrix} \dots & xy & \dots \\ & s & \end{pmatrix} \beta$$

$$\text{(GEXP)} \qquad \alpha \begin{pmatrix} \dots & xy & \dots \\ & s & \end{pmatrix} \beta \twoheadrightarrow_{TPG} \alpha \begin{pmatrix} \dots & x a^{(n+1)} b^{(n)} y & \dots \\ & s & \end{pmatrix} \beta.$$

Define the powers $s \twoheadrightarrow^k_{TPG} t$ and $s \twoheadrightarrow^*_{TPG} t$ as for PGs, and the language $\mathrm{L_G} = \{s |\ \mathbb{I}^* \twoheadrightarrow^*_{TPG} \begin{pmatrix} \mathrm{S} \\ s \end{pmatrix}\}$.[5]

[5] No TPG operation depends on the order of elements in a tuple, so why not let an expression be a set of typed strings, and let Mrg be simply set-union? The problem is that then the result of merging two occurrences of the same expression would just be the same expression again. Tuples avoid this kind of problem, allowing multiplicity of resources to be recognized. 'Multisets' or 'numerations' would do as well. Even if non-logical restrictions somehow rule out multiple identical resources in an expression, still it can be valuable to separate the logical structure from the external constraints as cleanly as possible. Chomsky (1995, p.244) worries that set-union is 'contradictory', but that is because he does not evaluate sets (or tuples) the way we do, with adjoint contraction 'checking and deleting' 'contradictory' features. Chomsky suggests expressions are sets and that merge maps sets S,T to {S,T} or $\{\gamma, \{\mathrm{S}, \mathrm{T}\}\}$ where γ is some function of S,T (Chomsky, 1995, pp.242-243). But this merge increases the syntactic complexity of constituents without bound, and unnecessarily – since complex constituents have much that cannot be syntactically relevant.

We introduce a 'performance restriction' on TPGs analogous to the earlier one for PGs. Let's extend our earlier notions so that a tuple is **proper** iff (i) every type in it has exactly one atom, (ii) no two types have the same atom, and (iii) GCON does not apply to it or to any result of applying Move to it. And a type it is **saturated** iff every type in it is saturated. A **minimalist TPG** (mTPG) is a TPG in which

$\mathbb{I}$ is rl and proper;

Mrg applies to a pair of tuples only if **both are proper** and **at least one of them is saturated**;

Move applies to a pair of typed strings in a tuple only if **both are proper** and **at least one of them is saturated**.

These restrictions extend the earlier ones, and so of course they are tentative. But note that, like the earlier restrictions, they do not add any new rules or devices, but simply restrict the lexical types and the domains of the rules. Keeping an eye on the comparisons we would like to make, we can observe that, as before, not only does a head combine with its arguments 'locally' before Mrg can apply to it, but the notion of being 'proper' has been extended in such a way that an element must be 'licensed' in the first available position before Mrg can apply again – giving us a kind of 'shortest move constraint' (Chomsky 1995, pp181ff) or 'relativized minimality' (Rizzi, 1990), where the domains are relativized by the classification of 'moving' expressions into types. The most revealing perspective on these constraints is provided by their roles in establishing the formal results in §4 below. But first some examples.

3.1. Example G_0 extended: wh-movement and islands.

Formal results are presented in §4 below, but it is easy to see, for example, that if relative clauses are formed by 'extraction' of a relative pronoun with a feature w, two constituents of this same type cannot be extracted at once. Ignoring the *who/whom* difference in (some varieties of) English for the moment, let's extend G_0 by adding w to $\mathbb{P}$ and 3 new lexical items:

$$\begin{pmatrix} \mathrm{D} & \mathrm{w} \\ \epsilon & \mathrm{who} \end{pmatrix} \begin{pmatrix} \mathrm{PD}^{\mathrm{l}}_{3\mathrm{s}} & \mathrm{ww}^{\mathrm{l}} \\ \epsilon & \mathrm{to} \end{pmatrix} \begin{pmatrix} \mathrm{w}^{\mathrm{r}}\mathrm{N}^{\mathrm{r}}\mathrm{NS}^{\mathrm{l}} \\ \epsilon \end{pmatrix}.$$

Intuitively, the first lexical item separates the D 'chain' into the empty part selected by the verb or preposition, and the pronounced part that moves to the wh-licensor.[6] The second lexical item allows 'pied piping'

[6] Capturing chains with a set of the linked elements is a natural idea inspired by Brody's work. I am grateful to Greg Kobele for useful discussions of this connection.

of a P phrase with a wh-object. And the third is the 'complementizer' that can license a wh-phrase to form a phrase that modifies a noun.

This grammar gets sentences like the following:

a girl praised a boy who$_i$ t_i laughs
a girl praised a boy who$_i$ t_i praised a girl who$_j$ t_j laughs.

But in spite of the many alternative derivations of many strings, there is no derivation at all for the following string, which would violate the 'complex NP' island constraint of Ross (1967):

* a boy who$_i$ a girl who$_j$ t_j praised t_i laughed.

That is, this string of words is not in L(G$_0$); remember again that the traces and indices are not part of the language. Similarly, the grammar generates

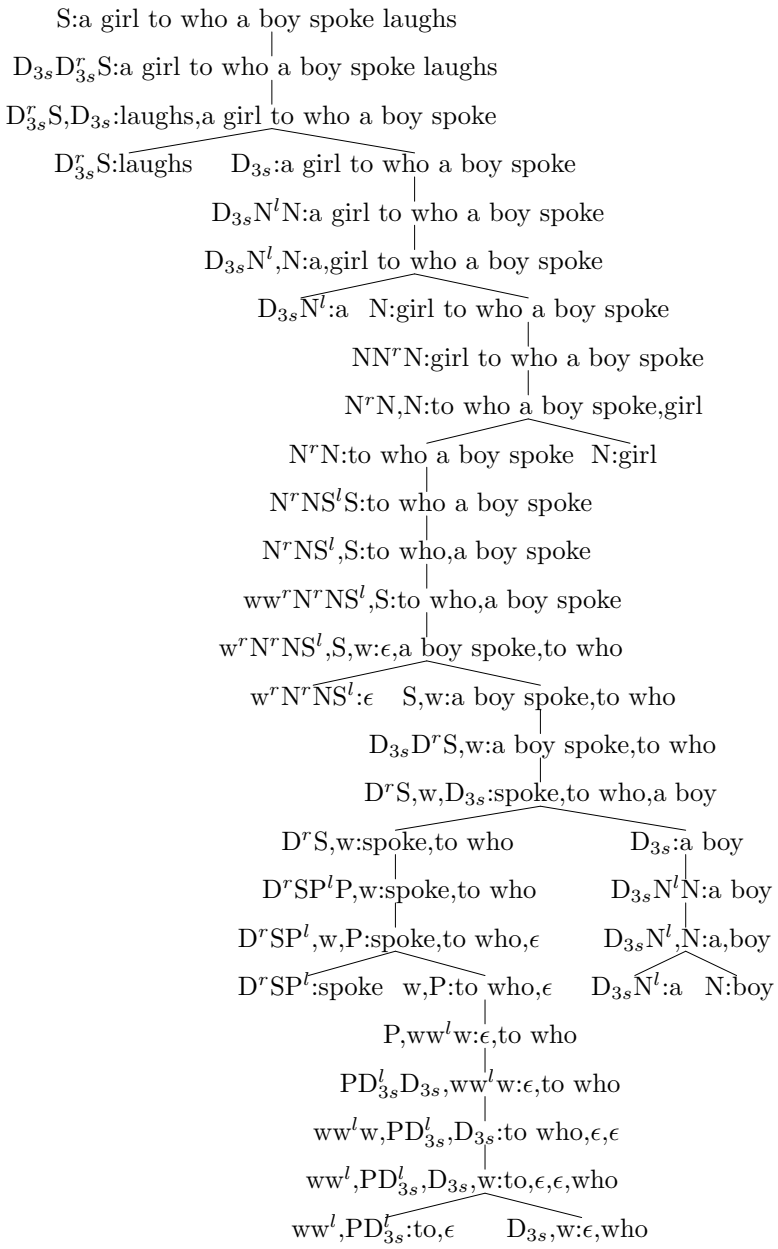
a girl spoke to a boy who$_i$ t_i laughs
a girl [to who]$_i$ a boy spoke t_i laughs
a girl who$_i$ a boy spoke to t_i laughs,

but not the island violations,

* a boy who$_i$ a girl to who$_j$ t_i spoke t_j laughs
* a boy to who$_i$ a girl who$_j$ t_j spoke t_i laughs.

Of course this tiny example is meant only to illustrate mTPG operation. It would interesting to consider how to handle the locality effects treated by recent proposals in Chomskian syntax (Fitzpatrick, 2002) and to compare alternative formal proposals about islands (Lambek, 2001; Fadda, 2002; Morrill, 2002; Lecomte and Retoré, 1999; Vermaat, 2004), but this is beyond the scope of this preliminary study.

In a standard derivation tree, binary branching indicates applications of (Mrg), unary branching indicates applications of (Move) or GCON, and each leaf is an element of the lexicon $\mathbb{I}$:



The proper typing and saturation requirements rule out many of the TPG derivation trees for this example; a computer implementation counts 6144. (Adding the prefix and feature-driven conditions of Proposition 1 below reduces the number to 8.)

3.2. Example G_1: non-configurational aux-2.

Some human languages with relatively free word order allow an auxiliary or other reduced element to appear only in 'second position'.[7] To illustrate the use of tupling to avoid order commitments, consider the following extremely simple grammar. Let the vocabulary $\Sigma = \{$Mary, John, self nom, acc, aux, praises$\}$, with $\mathbb{P} = \{\mathrm{s}, \mathrm{o}, \mathrm{v}, \mathrm{D}, \mathrm{R}, \mathrm{X}, \mathrm{S}\}$, and let $\leq$ be the reflexive transitive closure of the relation on $\mathbb{P}$

$$\mathrm{D} < \mathrm{R} \qquad \mathrm{o} < \mathrm{X} \qquad \mathrm{s} < \mathrm{X} \qquad \mathrm{v} < \mathrm{X}.$$

Let the lexicon $\mathbb{I}$ consist of the following 5 tuples:

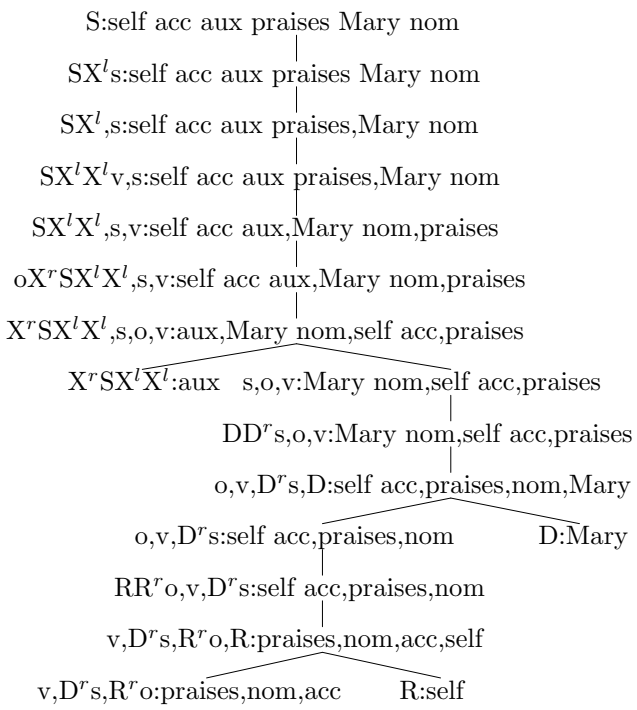
$$\begin{pmatrix} \mathrm{X}^{\mathrm{r}}\mathrm{S}\mathrm{X}^{\mathrm{l}}\mathrm{X}^{\mathrm{l}} \\ \text{aux} \end{pmatrix} \begin{pmatrix} \mathrm{v} & \mathrm{D}^{\mathrm{r}}\mathrm{s} & \mathrm{R}^{\mathrm{r}}\mathrm{o} \\ \text{praises} & \text{nom} & \text{acc} \end{pmatrix} \begin{pmatrix} \mathrm{D} \\ \text{John} \end{pmatrix} \begin{pmatrix} \mathrm{D} \\ \text{Mary} \end{pmatrix} \begin{pmatrix} \mathrm{R} \\ \text{self} \end{pmatrix}.$$

R is the category of determiners, including the reflexive *self*. Any element of R can be marked with accusative case (acc), but only non-reflexive D can be marked with nominative case (nom). Clearly $\mathrm{L}(\mathrm{G}_1)$ includes all six aux-2 orders:

Mary nom aux praises self acc	Mary nom aux self acc praises
self acc aux praises Mary nom	self acc aux Mary nom praises
praises aux Mary nom self acc	praises aux self acc Mary nom,

For example:

[7] E.g., Warlpiri, Pima, Papago (Munro, 1989; Hale, 1992; Smith, 2002).



A computer implementation counts 48 different mTPG derivation trees for this example (but only 6 with the prefix and feature-driven conditions of Proposition 1 below) and confirms that $\mathrm{L}(\mathrm{G}_1)$ does not include things like,

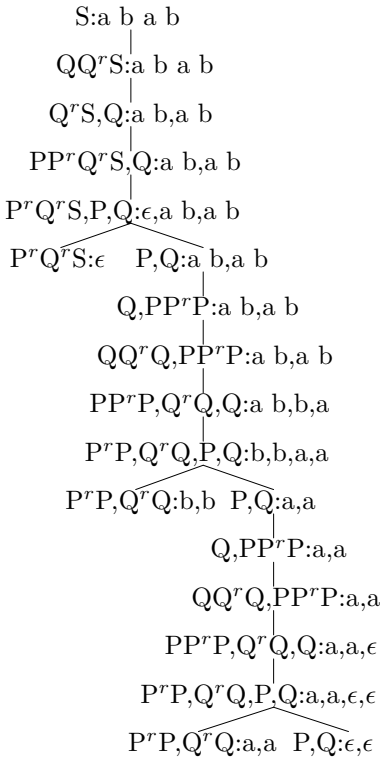
John acc aux self nom praises	self nom aux John acc praises
aux Mary nom praises John acc	Mary nom John acc aux praises
John acc praises Mary nom aux	John aux acc praises Mary nom.

3.3. Example G_2: crossing dependencies.

Let the vocabulary $\Sigma = \{\mathrm{a}, \mathrm{b}\}$, with the simple types $\mathbb{P} = \{\mathrm{P}, \mathrm{Q}, \mathrm{S}\}$ ordered by identity, with the following 4 tuples in the lexicon $\mathbb{I}$:

$$\begin{pmatrix} \mathrm{P} & \mathrm{Q} \\ \epsilon & \epsilon \end{pmatrix} \begin{pmatrix} \mathrm{P}^{\mathrm{r}}\mathrm{P} & \mathrm{Q}^{\mathrm{r}}\mathrm{Q} \\ \mathrm{a} & \mathrm{a} \end{pmatrix} \begin{pmatrix} \mathrm{P}^{\mathrm{r}}\mathrm{P} & \mathrm{Q}^{\mathrm{r}}\mathrm{Q} \\ \mathrm{b} & \mathrm{b} \end{pmatrix} \begin{pmatrix} \mathrm{P}^{\mathrm{r}}\mathrm{Q}^{\mathrm{r}}\mathrm{S} & 1 \\ \epsilon & \epsilon \end{pmatrix}.$$

Then the string abab $\in \mathrm{L}(\mathrm{G}_2)$ has a derivation like this:



$L(G_2) = \{xx|\ x \in \{a, b\}^*\}$ is a non-context-free language.[8]

4. The power of mTPGs

For $e = \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}$, let the **tuple type** of e, $\mathrm{tt}(e) = \langle t_1, \dots, t_k \rangle$. And for $S \subseteq \mathbb{E}$, $\mathrm{tt}(S) = \{\mathrm{tt}(e)|\ e \in S\}$. Let $\twoheadrightarrow^k_-$ be the results of restricting the mTPG relation $\twoheadrightarrow^k_{mTPG}$ to results obtained without GEXP. The following lemma provides the starting point for assessing the expressive power of mTPGs.

LEMMA 8. *For any mTPG G, the set* $\mathrm{tt}(\{e|\mathbb{I}^* \twoheadrightarrow^*_- e\})$ *of derivable tuple types is finite.*

[8] This grammar is simpler than the simplest MG grammar I know of that defines the same language (Cornell, 1996; Stabler, 1997). This grammar is not only conceptually more straightforward, but also smaller in the sense of requiring fewer symbols in its presentation.

Proof: Define

$$\begin{aligned}
\mathrm{A} &= \{\mathrm{p} \in \mathbb{P} |\ \mathrm{rpl} \in \mathrm{type}(\mathbb{I}) \text{ for } \mathrm{r}, \mathrm{l} \in \mathbb{T}^*\},\\
\mathrm{R} &= \{\mathrm{r} \in \mathbb{T}^* |\ \mathrm{rpl} \in \mathrm{type}(\mathbb{I}) \text{ for } \mathrm{p} \in \mathbb{P}, \mathrm{l} \in \mathbb{T}^*\},\\
\mathrm{L} &= \{\mathrm{l} \in \mathbb{T}^* |\ \mathrm{rpl} \in \mathrm{type}(\mathbb{I}) \text{ for } \mathrm{r} \in \mathbb{T}^*, \mathrm{p} \in \mathbb{P}\}.
\end{aligned}$$

Since the lexicon $\mathbb{I}$ is rl, R is a set of sequences of right adjoints and L is a set of sequences of left adjoints. Since the lexicon is finite, so are A, R, and L. Let s(R) be the set of suffixes of elements of R (including ϵ), and p(L) be the set of prefixes of elements of L (including ϵ). Since the lexicon is proper, GCON applies only to derived expressions, contracting an atom with a right adjoint at the beginning of some element of R or with a left adjoint at the end of some element of L, and since at least one of the arguments of Mrg and Move must be saturated, the type of every coordinate of every derivable tuple will be in the finite set:

$$\begin{aligned}
&\{\mathrm{rpl} |\ \mathrm{r} \in \mathrm{s(R)},\ \mathrm{p} \in \mathrm{A},\ \mathrm{l} \in \mathrm{p(L)}\} \cup\\
&\{\mathrm{qrpl} |\ \mathrm{r} \in \mathrm{s(R)},\ \mathrm{p}, \mathrm{q} \in \mathrm{A},\ \mathrm{l} \in \mathrm{p(L)}\} \cup\\
&\{\mathrm{rplq} |\ \mathrm{r} \in \mathrm{s(R)},\ \mathrm{p}, \mathrm{q} \in \mathrm{A},\ \mathrm{l} \in \mathrm{p(L)}\}.
\end{aligned}$$

So the lexical types fix a finite upper bound on the the size of the type that can appear in any coordinate in a derived expression, and the result follows if the number of coordinates in any derived expression is also bounded. It is, since only Mrg can produce a result with more coordinates than any of its arguments, but its arguments must be proper, so no tuple can have more than $2 * |\mathrm{A}|$ coordinates. □

For any mTPG grammar G, define the **relevant tuple types**,

$$\mathbb{TT}_G \subseteq \mathrm{tt}(\{e | \mathbb{I}^* \twoheadrightarrow^*_{\_} e\}) \subset \mathbb{T}^*_{\mathbb{P}}$$

as the tuple types of expressions that occur in derivations of elements of L_G.

We characterize the power of mTPGs by comparing them to MCFGs (Seki et al., 1991; Michaelis, 1998), using a MCFG variant, a 'normal form' introduced by Harkema (2001, Lemma 1). A **MCFG G** = $\langle \mathrm{N}, \mathrm{O}, \mathrm{F}, \mathrm{R}, \mathrm{S} \rangle$ such that:

i. N is a finite, non-empty set of non-terminal symbols.

ii. O is a set of i-tuples of strings, $i > 0$: $\mathrm{O} \subseteq \bigcup_{0<\mathrm{i}} (\Sigma^*)^{\mathrm{i}}$ for $\Sigma \neq \emptyset$ and $\Sigma \cap \mathrm{N} = \emptyset$.

iii. F is a finite set of functions from tuples of elements of O into O, $\mathrm{F} \subseteq \{g : \mathrm{O}^n \to \mathrm{O} |\ \text{any } n > 0\}$ meeting the following conditions.

   a. The dimensions of the domain and range of each $g \in \mathrm{F}$ are fixed in the following sense. For any $g : O^n \to O$ in F, there

are numbers $r(g) \geq 0$ and $d_i(g) \geq 0$ for $1 \leq i \leq n$ such that $g : (\Sigma^*)^{d_1(g)} \times \ldots \times (\Sigma^*)^{d_n} \to (\Sigma^*)^{r(g)}$.

b. The values of each $g : O^n \to O$ in F are fixed in the following sense. Let $X = \{x_{ij} |\ 1 \leq i \leq n, 1 \leq j \leq d_i(g)\}$ be a set of pairwise distinct variables and define $x_i = (x_{i1}, \ldots, x_{id_i(g)})$ for $1 \leq i \leq n$. Then for $1 \leq h \leq r(g)$ there are functions $g_h : \text{domain}(g) \to \Sigma^*$ such that for any $\theta \in \text{domain}(g)$, $g(\theta) = \langle g_1(\theta), \ldots, g_n(\theta)\rangle$ and each such g_h can be defined as concatenating components of its arguments as follows,

$$g_h(x_1, \ldots, x_n) = z_{h1} \ldots z_{hl_h(g)},$$

for some $l_h(g) \geq 0$ and some $z_{hl} \in X$, $1 \leq l \leq l_h(g)$.

c. Furthermore, the functions $g : O^n \to O$ in F are 'linear' (non-copying) and 'non-deleting' in the sense that for each pair (i, j), $1 \leq i \leq n$, $1 \leq j \leq d_i(g)$, there is exactly one h, $1 \leq h \leq r(g)$ and exactly one l, $1 \leq l \leq l_h(g)$ such that the variable z_{hl} in the definition above is the variable $x_{ij} \in X$.

iv. R is a finite set of 'rewrite rules' which pair n-ary rules of F with $n + 1$ nonterminals:

$$\mathrm{R} \subseteq \bigcup_{n \geq 0} (\{g \in \mathrm{F} | g : O^n \to O\} \times \mathrm{N}^{n+1}).$$

Usually we write any $(g, A, B_1, \ldots B_n) \in \mathrm{R}$ in the form: $A \to g[B_1, \ldots, B_n]$, unless $n = 0$ in which case $g \in O$ and we have a 'lexical rule' written $A \to g$. Furthermore we require:

a. Each $A \in \mathrm{N}$ has a dimension $d(A) \geq 0$ such that for any rule $A \to g[B_1, \ldots B_n]$, $g : O^n \to O$, $r(g) = d(A)$ and $d_i(g) = d(B_i)$ for all $1 \leq i \leq n$.

b. If $A \in N$ occurs in a lexical rule $A \to g$, then $d(A) = 1$.

c. There are no 'doublets': nonterminals on the right side of every rule $A \to g[B_1, \ldots B_n]$ are pairwise distinct.

v. $\mathrm{S} \in \mathrm{N}$ is the 'start' symbol, and $d(\mathrm{S}) = 1$.

Now, for each $A \in N$, define

$$\begin{aligned}
&\mathrm{L}^0_{\mathrm{G}}(\mathrm{A}) = \{\mathrm{g} | \mathrm{A} \to \mathrm{g}\},\\
&\mathrm{L}^{\mathrm{k+1}}_{\mathrm{G}}(\mathrm{A}) = \mathrm{L}^{\mathrm{k}}_{\mathrm{G}}(\mathrm{A}) \cup\\
&\qquad \{g(\theta_1 \ldots \theta_n) |\ A \to g[B_1, \ldots, B_n], \text{ and } \theta_i \in L^k_G(B_i),\ 1 \leq i \leq n\},\\
&\mathrm{L_G(A)} = \bigcup_{\mathrm{n} \geq 0} \mathrm{L}^{\mathrm{n}}_{\mathrm{G}}(\mathrm{A}).
\end{aligned}$$

So then we define $\mathrm{L_G} = \mathrm{L_G(S)}$.

Example MCFG$_2$. We show just the rules R, in which the vocabulary, nonterminals and string functions (defined below) can be identified. (For reasons that will become clear, the category names are either tuple types or subscripted typle types of example mTPG G$_2$.)

$$\begin{array}{l}
\langle \mathrm{P}, \mathrm{Q}, \epsilon, \epsilon\rangle_1 \rightarrow \epsilon \\
\langle \mathrm{P}, \mathrm{Q}, \epsilon, \epsilon\rangle_2 \rightarrow \epsilon \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{a}, \mathrm{a}\rangle_1 \rightarrow \mathrm{a} \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{a}, \mathrm{a}\rangle_2 \rightarrow \mathrm{a} \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{b}, \mathrm{b}\rangle_1 \rightarrow \mathrm{b} \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{b}, \mathrm{b}\rangle_2 \rightarrow \mathrm{b} \\
\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \epsilon\rangle_1 \rightarrow \epsilon
\end{array}$$

$$\begin{array}{l}
\langle \mathrm{P}, \mathrm{Q}\rangle \rightarrow \mathrm{t}_{1,2}[\langle \mathrm{P}, \mathrm{Q}, \epsilon, \epsilon\rangle_1\ \langle \mathrm{P}, \mathrm{Q}, \epsilon, \epsilon\rangle_2] \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}\rangle \rightarrow \mathrm{t}_{1,2}[\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{a}, \mathrm{a}\rangle_1\ \langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{a}, \mathrm{a}\rangle_2] \\
\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}\rangle \rightarrow \mathrm{t}_{1,2}[\langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{b}, \mathrm{b}\rangle_1\ \langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}, \mathrm{b}, \mathrm{b}\rangle_2] \\
\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}\rangle \rightarrow \mathrm{id}_1[\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \epsilon\rangle_1]
\end{array}$$

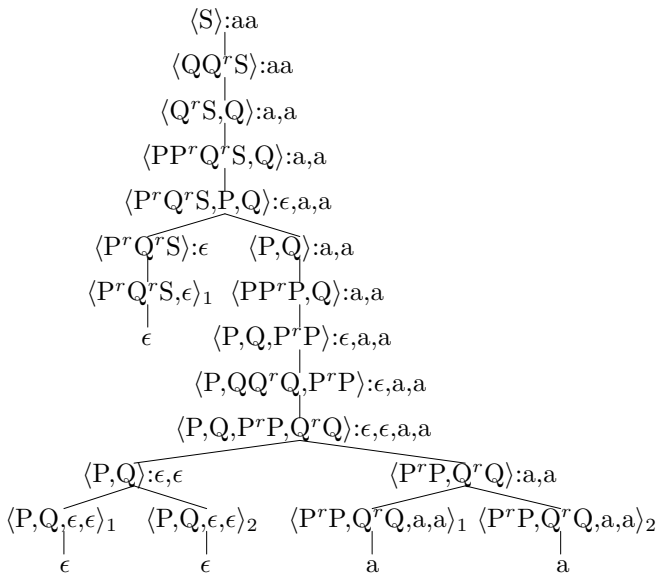
$$\begin{array}{l}
\langle \mathrm{S}\rangle \rightarrow \mathrm{id}_1[\langle \mathrm{Q}\mathrm{Q}^r\mathrm{S}\rangle] \\
\langle \mathrm{Q}\mathrm{Q}^r\mathrm{S}\rangle \rightarrow \mathrm{c}_{21}[\langle \mathrm{Q}^r\mathrm{S}, \mathrm{Q}\rangle] \\
\langle \mathrm{Q}^r\mathrm{S}, \mathrm{Q}\rangle \rightarrow \mathrm{id}_2[\langle \mathrm{P}\mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \mathrm{Q}\rangle] \\
\langle \mathrm{P}\mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \mathrm{Q}\rangle \rightarrow \mathrm{c}_{31,2}[\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \mathrm{P}, \mathrm{Q}\rangle] \\
\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}, \mathrm{P}, \mathrm{Q}\rangle \rightarrow \mathrm{t}_{1,23}[\langle \mathrm{P}^r\mathrm{Q}^r\mathrm{S}\rangle, \langle \mathrm{P}, \mathrm{Q}\rangle]
\end{array}$$

$$\begin{array}{l}
\langle \mathrm{P}, \mathrm{Q}\rangle \rightarrow \mathrm{id}_2[\langle \mathrm{P}\mathrm{P}^r\mathrm{P}, \mathrm{Q}\rangle] \\
\langle \mathrm{P}\mathrm{P}^r\mathrm{P}, \mathrm{Q}\rangle \rightarrow \mathrm{c}_{31,2}[\langle \mathrm{P}, \mathrm{Q}, \mathrm{P}^r\mathrm{P}\rangle] \\
\langle \mathrm{P}, \mathrm{Q}, \mathrm{P}^r\mathrm{P}\rangle \rightarrow \mathrm{id}_3[\langle \mathrm{P}, \mathrm{Q}\mathrm{Q}^r\mathrm{Q}, \mathrm{P}^r\mathrm{P}\rangle] \\
\langle \mathrm{P}, \mathrm{Q}\mathrm{Q}^r\mathrm{Q}, \mathrm{P}^r\mathrm{P}\rangle \rightarrow \mathrm{c}_{1,42,3}[\langle \mathrm{P}, \mathrm{Q}, \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}\rangle] \\
\langle \mathrm{P}, \mathrm{Q}, \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}\rangle \rightarrow \mathrm{t}_{12,34}[\langle \mathrm{P}, \mathrm{Q}\rangle, \langle \mathrm{P}^r\mathrm{P}, \mathrm{Q}^r\mathrm{Q}\rangle]
\end{array}$$

These rules use string functions which can be defined as follows:

$$\begin{array}{l}
\mathrm{t}_{1,2}(s,t) = \langle s,t\rangle \\
\mathrm{t}_{1,23}(s,\langle t,u\rangle) = \langle s,t,u\rangle \\
\mathrm{t}_{12,34}(\langle s,t\rangle,\langle u,v\rangle) = \langle s,t,u,v\rangle \\
\mathrm{id}_1(s) = \langle s\rangle \\
\mathrm{id}_2(s,t) = \langle s,t\rangle \\
\mathrm{id}_3(s,t,u) = \langle s,t,u\rangle \\
\mathrm{c}_{21}(s,t) = \langle ts\rangle \\
\mathrm{c}_{31,2}(s,t,u) = \langle us,t\rangle \\
\mathrm{c}_{1,42,3}(s,t,u,v) = \langle s,vt,u\rangle
\end{array}$$

With these definitions, L(MCFG$_2$)=$\{xx|\ x \in \{\mathrm{a},\mathrm{b}\}^*\}$, with derivations like this:



Except for the special treatment of lexical items (in order to allow that every lexical category in the MCFG has dimension 1), this derivation tree is isomorphic to an mTPG derivation. So it is easy to show:

LEMMA 9. *Every mTPG language is MCFG definable.*

Proof: For any mTPG $\mathrm{M} = \langle \Sigma, \mathbb{P}, \leq, \mathbb{I}, \mathrm{S} \rangle$, define MCFG $\mathrm{G_M}$ as the smallest grammar with the following rules, with the nonterminals N, string tuples O, and string functions as specified:

i. For each $\begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix} \in \mathbb{I}$ we have the following rules:

$$\langle t_1, \dots, t_k, s_1, \dots, s_k \rangle_1 \to s_1$$
$$\dots$$
$$\langle t_1, \dots, t_k, s_1, \dots, s_k \rangle_k \to s_k$$
$$\langle t_1, \dots, t_k \rangle \to \mathrm{t}_{1k}[\langle t_1, \dots, t_k, s_1, \dots, s_k \rangle_1, \dots, \langle t_1, \dots, t_k, s_1, \dots, s_k \rangle_k]$$

where the string function in the last rule is defined as follows:

$$\mathrm{t}_{1k}(s_1, \dots, s_k) = \langle s_1, \dots, s_k \rangle.$$

ii. For each $\langle t_1, \dots, t_i \rangle, \langle t_{i+1}, \dots, t_k \rangle \in \mathbb{TT}_G$ such that $e_1 \bullet e_2 \in \mathbb{TT}_G$, we have the rule

$$\langle t_1, \dots, t_k \rangle \to \mathrm{t}_{1 \dots t_i, t_{i+1} \dots k}[\langle t_1, \dots, t_i \rangle, \langle t_{i+1}, \dots, t_k \rangle]$$

where the string function is defined as follows:

$$\mathrm{t}_{1 \dots t_i, t_{i+1} \dots k}(\langle s_1, \dots, s_i \rangle, \langle s_{i+1}, \dots, s_k \rangle) = \langle s_1, \dots, s_k \rangle.$$

iii. For each $\langle t_1, \ldots, t_k \rangle \in \mathbb{TT}_G$ such that $\langle t_i t_j \rangle \bullet \langle t_1, \ldots, t_k \rangle_{-i-j} \in \mathbb{TT}_G$, we have the rule

$$\langle t_i t_j \rangle \bullet \langle t_1, \ldots, t_k \rangle_{-i-j} \rightarrow \mathrm{c}_{ij,1\ldots t_{i-2}}[\langle t_1, \ldots, t_k \rangle]$$

where the string function is defined as follows:

$$\mathrm{c}_{ij,1\ldots t_{i-2}}(s_1, \ldots, s_i) = \langle s_i s_j \rangle \bullet \langle s_1, \ldots, s_i \rangle_{-i-j}.$$

iv. For each $\langle t_1, \ldots, t_k \rangle \in \mathbb{TT}_G$ such that $t_i = x a^{(n)} b^{(n+1)} y$, either $a \leq b$ and n is even, or $b \leq a$ and n is odd, and $\langle t_1, \ldots, t_{i-i}, xy, t_{i+1}, \ldots, t_k \rangle \in \mathbb{TT}_G$ we have the rule:

$$\langle t_1, \ldots, t_{i-i}, xy, t_{i+1}, \ldots, t_k \rangle \rightarrow \mathrm{id}_k[\langle t_1, \ldots, t_k \rangle]$$

where id_k is the identity on k-tuples of strings.

Since $\mathbb{I}$ and $\mathbb{TT}_G$ are finite, so is the set of rules defined by the previous 4 clauses. We establish the lemma by showing that for $\langle t_1, \ldots, t_k \rangle \in \mathbb{TT}_G$, $\langle s_1, \ldots, s_k \rangle \in L_G(\langle t_1, \ldots, t_k \rangle)$ iff $\mathbb{I}^* \rightarrow_-^* \begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix}$.

($\Rightarrow$) We use an induction to show that for all $n \geq 0$, $\langle s_1, \ldots, s_k \rangle \in L_G^n(\langle t_1, \ldots, t_k \rangle)$ implies $\mathbb{I}^* \rightarrow_-^* \begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix}$. By the definition of $\mathrm{G_M}$, it is clear that when $n = 0$, L_G^n has no elements whose category is a tuple type, and so the result holds vacuously. So consider,

($n = 1$) By clause i of the definition of $\mathrm{G_M}$, $\langle s_1, \ldots, s_k \rangle \in L_G^1(\langle t_1, \ldots, t_k \rangle)$ if $\begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix} \in \mathbb{I}$. And nothing else with a category $\langle t_1, \ldots, t_k \rangle \in \mathbb{TT}_G$ is in L_G^1 since the rules introduced by clauses ii-iv act only on constituents with categories in $\mathbb{TT}_G$, so the result holds.

(IH) for $m \leq n$, $\langle s_1, \ldots, s_k \rangle \in L_G^m(\langle t_1, \ldots, t_k \rangle)$ implies $\mathbb{I}^* \rightarrow_-^* \begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix}$.

Now assume $\langle s_1, \ldots, s_k \rangle \in L_G^{n+1}(\langle t_1, \ldots, t_k \rangle)$ for $n > 0$. Then

$$\langle s_1, \ldots, s_k \rangle = g(\theta_1, \ldots, \theta_i),$$

where

$$\langle t_1, \ldots, t_k \rangle \rightarrow g[B_1 \ldots B_i] \text{ for } 1 \leq j \leq i,\ \theta_j \in L_G^n(B_j).$$

This rule must have been introduced by one of the clauses ii-iv in the definition of $\mathrm{G_M}$, and it is easy to use IH and one application of Mrg, Move or GCON (respectively) to see that the result holds.

($\Leftarrow$) A similarly simple induction shows that for all $n \geq 0$, $\mathbb{I}^* \rightarrow_-^n \begin{pmatrix} t_1 & \ldots & t_k \\ s_1 & \ldots & s_k \end{pmatrix}$ implies $\langle s_1, \ldots, s_k \rangle \in L_G(\langle t_1, \ldots, t_k \rangle)$.

($n = 0$) In this case, $\begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix} \in \mathbb{I}$ and so the result holds by clause i of the definition of $\mathrm{G_M}$.
(IH) for $m \leq n$, $\mathbb{I}^* \twoheadrightarrow^m_- \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}$ implies $\langle s_1, \dots, s_k \rangle \in L_G(\langle t_1, \dots, t_k \rangle)$.
Now assume $\mathbb{I}^* \twoheadrightarrow^{n+1}_- \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix}$. In this case, Mrg, Move or GCON applies to expressions for which we have IH to guarantee the existence of a corresponding MCFG derivation which can be extended by one application of a rule introduced by ii, iii or iv (respectively) to establish the result. □

We can also establish the converse result. We first provide an example to illustrate the basic idea.

Example. Consider an MCFG with the following rules:

$$\begin{array}{l} S \twoheadrightarrow \mathrm{g}[A] \\ A \twoheadrightarrow \mathrm{h}[B, C, D, E] \\ B \twoheadrightarrow a \\ C \twoheadrightarrow b \\ D \twoheadrightarrow c \\ E \twoheadrightarrow d \end{array}$$

where the string functions are defined as follows:

$$\begin{array}{l} \mathrm{g}(s, t, u, v) = \langle sv, ut \rangle \\ \mathrm{h}(s, t) = \langle st \rangle \end{array}$$

Then $\mathrm{L_G} = \{\mathrm{adcb}\}$. We construct a mTPG which defines the same language in a similar way. We have an mTPG lexical item for each lexical productions,

$$\begin{pmatrix} B \\ a \end{pmatrix} \qquad \begin{pmatrix} C \\ b \end{pmatrix} \qquad \begin{pmatrix} D \\ c \end{pmatrix} \qquad \begin{pmatrix} E \\ d \end{pmatrix}.$$

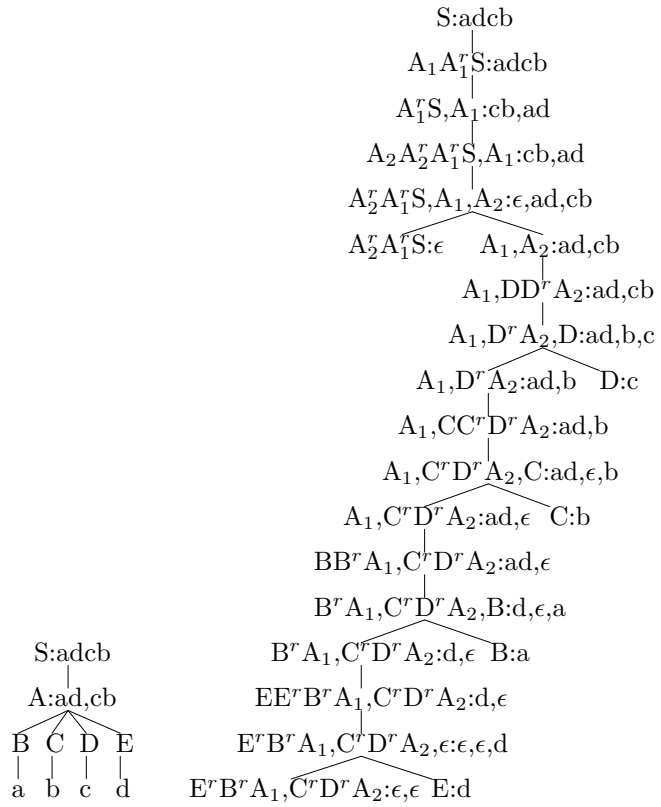
Corresponding to the rule $A \twoheadrightarrow \mathrm{h}[B_1, B_2, B_3, B_4]$, we introduce an mTPG lexical item which will properly build the components of A according to the definition of h,

$$\begin{pmatrix} B^r E^r A_1 & C^r D^r A_2 \\ \epsilon & \epsilon \end{pmatrix}.$$

And finally, corresponding to the rule $S \twoheadrightarrow \mathrm{g}[A]$ we introduce the lexical item

$$\begin{pmatrix} A_2^r A_1^r S \\ \epsilon \end{pmatrix}.$$

Now, corresponding to a MCFG derivation like the one on the left below, we have the mTPG derivation on the right:



These derivation trees are certainly not isomorphic, but the correspondence between them is still fairly simple: the non-lexical MCFG rules correspond to empty mTPG lexical items, and a succession of merges and contractions constructs each component of each derived category. Comparing the derivations above, notice the direct mTPG representation of (i) the MCFG lexical items, (ii) the two components of the derived A, and (iii) the unique component of the derived S.

LEMMA 10. *Every MCFG language is mTPG definable.*

Proof: Consider any MCFG $G = \langle N, O, F, R, S\rangle$. We define an mTPG grammar M with the following 2 clauses that specify its lexicon, from which its types and vocabulary can be inferred:

i. For each lexical rule $A \rightarrow g$,

$$\begin{pmatrix} A_1 \\ g \end{pmatrix} \in \mathbb{I}.$$

ii. For each non-lexical rule $A \rightarrow g[B1, \ldots, Bn]$

$$\begin{pmatrix} x_1 A_1 & \ldots & x_{d(A)} A_{d(A)} \\ \epsilon & \ldots & \epsilon \end{pmatrix} \in \mathbb{I},$$

where for $1 \leq i \leq d(A)$, $x_i = Bi1^r_{j_1} \dots Bik^r_{j_k}$ where by the definition of g, the i'th component of its value is the concatenation of the j_k'th component of its i_k'th argument with... with the j_1'th component of its i_1'th argument.

So $\mathbb{I}$ has $|R|$ elements. We establish $L_G = L_M$ by showing that for all A $\in$ N, $\langle s_1, \dots, s_{d(A)} \rangle \in L_G(A)$ iff $\mathbb{I}^* \twoheadrightarrow^*_- \begin{pmatrix} A_1 & \dots & A_{d(A)} \\ s_1 & \dots & s_{d(A)} \end{pmatrix}$.

($\Rightarrow$) We use an induction to show that for all $n \geq 0$, $\langle s_1, \dots, s_k \rangle \in L^n_G(A)$ implies $\mathbb{I}^* \twoheadrightarrow^*_- \begin{pmatrix} A_1 & \dots & A_k \\ s_1 & \dots & s_k \end{pmatrix}$.

($n = 0$) By the definition of MCFGs, all lexical categories have dimension 1, and so in this case $k = 1$, and by clause i of the definition of M, $\begin{pmatrix} A_1 \\ s_1 \end{pmatrix} \in \mathbb{I}$.

(IH) for $m \leq n$, $\langle s_1, \dots, s_k \rangle \in L^m_G(A)$ implies $\mathbb{I}^* \twoheadrightarrow^*_- \begin{pmatrix} A_1 & \dots & A_k \\ s_1 & \dots & s_k \end{pmatrix}$.

Now assume $\langle s_1, \dots, s_k \rangle \in L^{n+1}_G(A)$. Then

$$\langle s_1, \dots, s_k \rangle = g(\theta_1, \dots, \theta_i),$$

where

$$A \to g[B_1 \dots B_i] \text{ for } 1 \leq j \leq i,\ \theta_j \in L^n_G(B_j).$$

For each B_j and θ_j (for $1 \leq j \leq i$), the IH guarantees that we have mTPG derivations of $\begin{pmatrix} (B_j)_1 & \dots & (B_j)_{d(B_j)} \\ (s_j)_1 & \dots & (s_j)_{d(B_j)} \end{pmatrix}$. And by clause ii of the definition of M and the definition of mTPG operations, the components of A will be properly assembled by merging these mTPG results and performing all possible contractions. The 'no doublet' condition in the definition of MCFGs ensures that each component will be unambiguously named. And clause ii of the definition of M, with the 'linearity' and 'non-deleting' conditions on MCFGs, guarantees that each such element will be placed in exactly the position defined by g.

($\Leftarrow$) We use an induction again to show that for all A $\in$ N and all $n \geq 0$, $\mathbb{I}^* \twoheadrightarrow^n_- \begin{pmatrix} A_1 & \dots & A_k \\ s_1 & \dots & s_k \end{pmatrix}$ implies $\langle s_1, \dots, s_k \rangle \in L_G(A)$.

($n = 0$) In this case, $\begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \end{pmatrix} \in \mathbb{I}$ and so the result holds by clause i of the definition of M.

(IH) for $m \leq n$, $\mathbb{I}^* \twoheadrightarrow^m_- \begin{pmatrix} A_1 & \dots & A_k \\ s_1 & \dots & s_k \end{pmatrix}$ implies $\langle s_1, \dots, s_k \rangle \in L_G(A)$.

Now assume $\mathbb{I}^* \twoheadrightarrow^{n+1}_- \begin{pmatrix} A_1 & \dots & A_k \\ s_1 & \dots & s_k \end{pmatrix}$ for some A $\in$ N. In this case, Mrg, Move or GCON derive this expression and by the definition of M, it

can only be from a lexical item $\begin{pmatrix} x_1 A_1 & \dots & x_{d(A)} A_{d(A)} \\ \epsilon & \dots & \epsilon \end{pmatrix}$ introduced by clause ii, together with constituents (proper, unsaturated constituents, by the definition of mTPG) that contract with all of the x_j (for $1 \leq j \leq d(A)$). By clause ii, these contractions must assemble the components of $B1, \dots, Bn$ for some $A \to g[B1, \dots, Bn]$ in G, and for each of these, the IH guarantees that the components $\langle t_1, \dots, t_{d(Bi)} \rangle \in L_G(Bi)$. The definition of the x_j guarantees that these components are assembled according to g so that $\langle s_1, \dots, s_k \rangle \in L_G(A)$. □

THEOREM 3. *Language L is mTPG-definable iff it is MCFG-definable.*

Proof: Immediate from the previous two lemmas. □

Note that mTPG derivations are not all minimal in size. For example, after applying move to two components to produce a result that can contract, we can apply Move again to to different components before contracting. We can add a requirement to Move in order to block this. Let's say a derivation is **feature-driven** iff every Move step is immediately followed by a GCON step that contracts the new type formed by that step. We can restrict things further by requiring derivations to be **prefix** form in the sense that the second argument of Mrg is always saturated. The following 'normal form' result is straightforward:

PROPOSITION 1. *For rl expressions $e_1, \dots, e_n$ and atomic a, $e_1 \dots e_n \twoheadrightarrow^*_{\text{mTPG}} a$ iff it has a prefix, feature-driven mTPG derivation.*

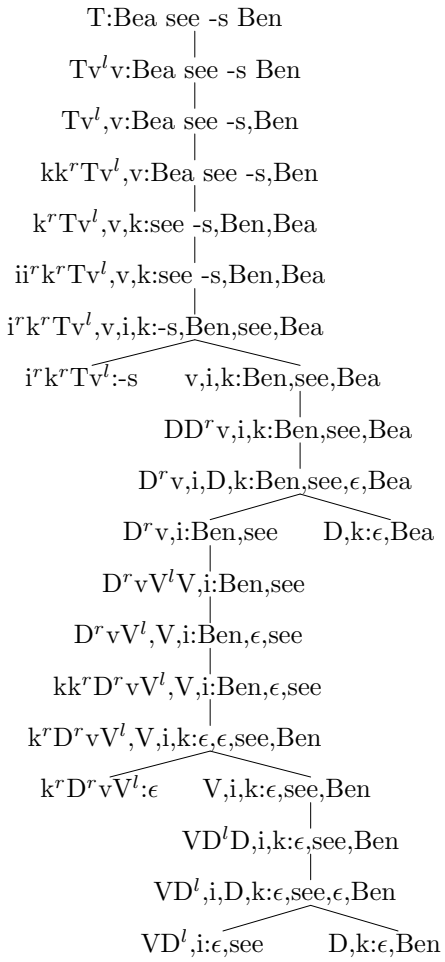
5. More challenging phenomena

5.1. Example G_3: remnant movement.

Recently, some linguists have proposed analyses in which a phrase from which something has been extracted – a 'remnant' – can move (Kayne, 1994; Koopman and Szabolcsi, 2000). This was impossible in some earlier theories of movement. The following mTPG is inspired by Mahajan (2000), with $\leq$ the identity:

$$\begin{pmatrix} \text{D} & \text{k} \\ \epsilon & \text{Bea} \end{pmatrix} \begin{pmatrix} \text{D} & \text{k} \\ \epsilon & \text{Ben} \end{pmatrix} \begin{pmatrix} \text{VD}^{\text{l}} & \text{i} \\ \epsilon & \text{see} \end{pmatrix} \begin{pmatrix} \text{i}^{\text{r}}\text{k}^{\text{r}}\text{Tv}^{\text{l}} \\ \text{-s} \end{pmatrix} \begin{pmatrix} \text{k}^{\text{r}}\text{D}^{\text{r}}\text{vV}^{\text{l}} \\ \epsilon \end{pmatrix}.$$

The idea here is that 'determiner phrases' like *Bea* and *Ben* move to have their 'case' feature k checked, so the object of a verb phrase is extracted for case by the (empty) transitivizing v, and then the verb phrase remnant moves to have its 'inflection' feature i checked. We see this in the following derivation, with start type T:



5.2. Example G_4: head movement and affix hopping.

Lambek (1999) presents an analysis of English auxiliaries and tense[9] but does not treat the regular affixes as part of the syntax, as suggested by Chomsky (1956) and much following work. Many recent Chomskian proposals assume that the tense and aspectual affixes are placed by some kind of 'head movement', and sometimes also by 'affix hopping'. A simplistic formulation of this idea is defined in Stabler (2001), and we can now get a similar analysis with mTPGs like the following. We use start category C, and the familiar categories T (tense), N (noun), D (determiner), M (modal), H (have), B (be), plus the features w (wh-requirement), k (case requirement), q (question movement requirement), t (requiring tense affix), e (requiring affix -en), i (requiring affix

[9] Cf. also categorial unification grammars: Bach (1983) and others.

-en or -ing), x (requiring adjunction to little v for external argument), and h (needs affix hop):

$$\begin{pmatrix} \text{C}^{\text{l}} \\ \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{q}^{\text{r}}\text{C}^{\text{l}} \\ \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{q}^{\text{r}}\text{w}^{\text{r}}\text{C}^{\text{l}} \\ \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{kN}^{\text{l}} & \text{D} \\ \text{the} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{k} & \text{D} \\ \text{he} & \epsilon \end{pmatrix}$$

$$\begin{pmatrix} \text{wN}^{\text{l}} & \text{k} & \text{D} \\ \text{which} & \epsilon & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{N} \\ \text{king} \end{pmatrix} \quad \begin{pmatrix} \text{N} \\ \text{pie} \end{pmatrix} \quad \begin{pmatrix} \text{x} & \text{k}^{\text{r}}\text{VD}^{\text{l}} \\ \text{eat} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{x} & \text{V} \\ \text{laugh} & \epsilon \end{pmatrix}$$

$$\begin{pmatrix} \text{h}^{\text{r}}\text{k}^{\text{r}}\text{Tv}^{\text{l}} \\ \text{-s} \end{pmatrix} \quad \begin{pmatrix} \text{x}^{\text{r}}\text{h} & \text{D}^{\text{r}}\text{vv}^{\text{l}} \\ \epsilon & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{t}^{\text{r}}\text{k}^{\text{r}}\text{TM}^{\text{l}} \\ \text{-s} \end{pmatrix} \quad \begin{pmatrix} \text{t}^{\text{r}}\text{k}^{\text{r}}\text{Tv}^{\text{l}} \\ \text{-s} \end{pmatrix} \quad \begin{pmatrix} \text{t}^{\text{r}}\text{q} & \text{k}^{\text{r}}\text{TM}^{\text{l}} \\ \text{-s} & \epsilon \end{pmatrix}$$

$$\begin{pmatrix} \text{t} & \text{MH}^{\text{l}} \\ \text{will} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{t} & \text{Mv}^{\text{l}} \\ \text{will} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{HE}^{\text{l}} \\ \text{have} \end{pmatrix} \quad \begin{pmatrix} \text{t} & \text{HE}^{\text{l}} \\ \text{have} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{e}^{\text{r}}\text{EB}^{\text{l}} \\ \text{-en} \end{pmatrix}$$

$$\begin{pmatrix} \text{e}^{\text{r}}\text{Ev}^{\text{l}} \\ \text{-en} \end{pmatrix} \quad \begin{pmatrix} \text{BI}^{\text{l}} \\ \text{be} \end{pmatrix} \quad \begin{pmatrix} \text{e} & \text{BI}^{\text{l}} \\ \text{be} & \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{i}^{\text{r}}\text{Iv}^{\text{l}} \\ \text{-ing} \end{pmatrix} \quad \begin{pmatrix} \text{x}^{\text{r}}\text{D}^{\text{r}}\text{vv}^{\text{l}} \\ \epsilon \end{pmatrix} \quad \begin{pmatrix} \text{x}^{\text{r}}\text{i} & \text{D}^{\text{r}}\text{vv}^{\text{l}} \\ \epsilon & \epsilon \end{pmatrix},$$

with $\leq$ the reflexive, transitive closure of $\text{B} < \text{H} < \text{M}$ and $\text{i} < \text{e} < \text{t}$. This grammar mimics a traditional Chomskian analysis, and is English-like on a range of constructions:

the king eat -s the pie	the king will -s eat the pie
*eat -s the king the pie	will -s the king eat the pie
the king be -s eat -ing the pie	the king have -s eat -en the pie
be -s the king eat -ing the pie	have -s the king eat -en the pie
*eat -ing the king be -s the pie	*eat -en the king have -s the pie

the king will -s have be -en eat -ing the pie
which pie will -s the king have be -en eat -ing
*which pie have -s the king will be eat -ing

We have the following derivation, for example:

As complex as it is, this analysis is very much simpler than the one in Stabler (2001) which requires many special generating functions. It would be interesting to simplify it further and to try cover the

range of data treated by more serious recent proposals in this tradition (Rizzi 1990, pp.20-24; Roberts 1993, §3.2.1; Lasnik 1995; Culicover 1999, §§3.4,3.6.3; Merchant 2003).

5.3. Partial wh-movement and successive cyclicity.

When wh-phrases move out of an embedded clause, it is often assumed that they move through an intervening empy operator position of some kind. Some languages like German, Hindi and Hungarian allow constructions in which the matrix clause has a reduced wh-element and an intervening operator position holds the regular question word (Fanselow and Mahajan, 2000; Horvath, 2000):

Was$_j$ hast Du t_j geglaubt [wen$_i$ sie t_i gesehen hat]$_j$?
what have you thought who she seen has

'who do you think she has seen'

There is a stage at which similar constructions seem natural even to children learning English (Thornton, 1990; McDaniel et al., 1995). There are many ways to introduce wh-associates like this in mTPGs. To quickly illustrate one kind of option, we extend the previous grammar G_4 to get:

What have -s he think -en who he have -s see -en,

by adding the following lexical items (introducing the new simple type O for the embedded 'operator' position):

$$\begin{pmatrix} \text{x} & \text{k}^\text{r}\text{VD}^\text{l} \\ \text{see} & \epsilon \end{pmatrix} \begin{pmatrix} \text{x} & \text{VC}^\text{l} \\ \text{think} & \epsilon \end{pmatrix} \begin{pmatrix} \text{w} & \text{k} & \text{D} \\ \text{who} & \epsilon & \epsilon \end{pmatrix} \begin{pmatrix} \text{w} & \text{CO}^\text{l} \\ \text{what} & \epsilon \end{pmatrix} \begin{pmatrix} \text{w}^\text{r}\text{OT}^\text{l} \\ \epsilon \end{pmatrix}.$$

Various kinds of 'domain registration' phenomena can be handled with methods like this. When the domain is not overtly registered – when there is no phonetically overt intermediate element – we may nevertheless want intervening positions ('traces') for successive cyclic wh-movement, as would result from empty versions of these lexical items for O and C.

6. Interpreting mTPGs

We quickly sketch a first, simple idea for specifying denotations of mTPG expressions. Moortgat (2003) points out that the associativity of pregroup types has the nice effect of avoiding overcommitment to the order in which arguments are provided to the verb, and this should be mirrored in the semantics (cf. Vermeulen and Visser; Kracht). We

specify meanings with a higher order logic L (cf. Church's simple theory of types).The denotation of a sequence of n terms (a type) is a sequence of n semantic values, represented by logical forms. So a k-tuple of types in the syntax denotes a k-tuple of sequences of values in the semantic domain. By Proposition 1, any mTPG result has an feature-driven derivation, so we can compose Move and GCON to get MCON. And when two terms are contracted, we unify their logical forms.

imTPG expressions are elements of $\mathbb{E} = \bigcup_{n\geq 0} (\mathbb{R}^n_{\mathbb{P}} \times (\Sigma \cup \{\epsilon\})^* \times \mathrm{L}^n)^*$, written $\begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \\ i_1 & \dots & i_k \end{pmatrix}$, or $t_1, \dots, t_k : s_1, \dots, s_k : i_1, \dots, i_k$. A **imTPG** $\mathrm{G} = \langle \Sigma, \mathbb{P}, \leq, \mathrm{L}, \mathbb{I}, \mathrm{S} \rangle$, now with $\mathrm{rl}\,\mathbb{I} \subset \bigcup_{n\geq 0} (\mathbb{R}^n_{\mathbb{P}} \times (\Sigma \cup \{\epsilon\}) \times \mathrm{L}^n)^*$. $\twoheadrightarrow^*_{imTPG}$ is the reflexive, transitive closure of the relation on $\mathbb{E}^*$ defined by Mrg and MCON as follows. For any proper tuples e_1, e_2, with at least one saturated, and any sequences of tuples α, β,

$$\text{(Mrg)} \quad \alpha\ e_1\ e_2\ \beta \twoheadrightarrow_{imTPG} \alpha\ e_1 \bullet e_2\ \beta.$$

Where $k > 1, 1 \leq i \leq k, 1 \leq j < k$, and t_i, t_j are proper, and at least one is saturated, and where

$$\theta \in \mathrm{mgu}(i_i, i_j) \qquad t_i = xa^{(n)} \qquad t_j = b^{(n+1)}y,$$

and either $a \leq b$ and n is even, or $b \leq a$ and n is odd:

$$\text{(MCON)} \quad \alpha \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \\ i_1 & \dots & i_k \end{pmatrix} \beta \twoheadrightarrow_{\mathrm{imTPG}} \alpha \left(\begin{pmatrix} xy \\ s_i s_j \\ i_i \end{pmatrix} \bullet \begin{pmatrix} t_1 & \dots & t_k \\ s_1 & \dots & s_k \\ i_1 & \dots & i_k \end{pmatrix}_{-\mathrm{i}-\mathrm{j}} \right) \theta\ \beta.$$

Note that we write substitution θ in postfix position – it applies to the whole tuple that MCON creates, possibly affecting all the logical forms in it. And recall that by the definition of most general unifier (mgu), $i_i\theta = i_j\theta$. See for example Huet (1975) and Nadathur and Miller (1998) on computational issues in higher order unification.

Example iG_0**,** adding semantic coordinates to lexical items of G_0. If there is more than one logical form in a sequence, in a single coordinate,

we separate them with dots for readability.

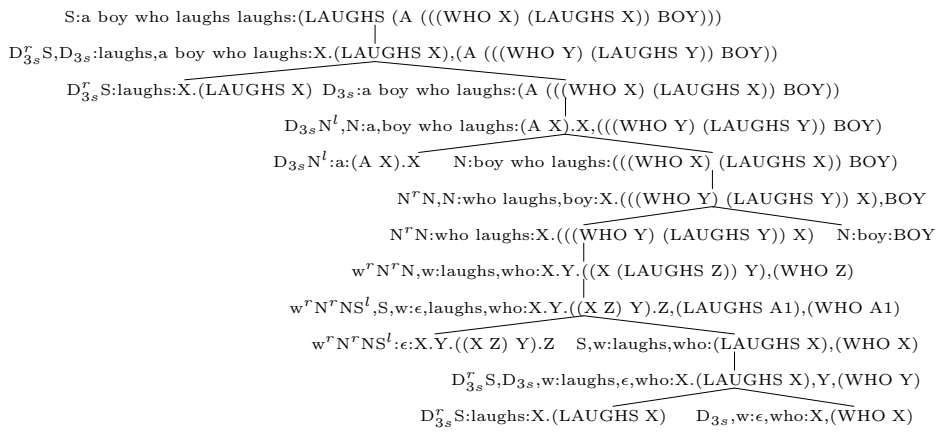
$$\begin{pmatrix} D_{3s}N^{l} \\ a \\ (A\ X).X \end{pmatrix} \quad \begin{pmatrix} N \\ boy \\ BOY \end{pmatrix} \quad \begin{pmatrix} N \\ girl \\ GIRL \end{pmatrix}$$

$$\begin{pmatrix} D^{r}_{3s}S \\ laughs \\ X.(X\ LAUGHS) \end{pmatrix} \quad \begin{pmatrix} D^{r}SD^{l} \\ praised \\ X.((PRAISED\ Y)X).Y \end{pmatrix} \quad \begin{pmatrix} D^{r}SP^{l} \\ spoke \\ X.((SPOKE\ Y)\ X).Y \end{pmatrix}$$

$$\begin{pmatrix} PD^{l} \\ to \\ (TO\ X).X \end{pmatrix} \quad \begin{pmatrix} D \\ we \\ WE \end{pmatrix} \quad \begin{pmatrix} D^{r}DD^{l} \\ and \\ X.((AND\ Y)\ X).Y \end{pmatrix}$$

$$\begin{pmatrix} D_{3s} & w \\ \epsilon & who \\ X & (WHO\ X) \end{pmatrix} \quad \begin{pmatrix} w^{r}N^{r}NS^{l} \\ \epsilon \\ X.Y.((X\ Z)\ Y).Z \end{pmatrix} \quad \begin{pmatrix} ww^{l} & PD^{l}_{3s} \\ to & \epsilon \\ X.X & (TO\ Y).Y \end{pmatrix}$$

Note that the interpretation of every adjoint is a variable (the scope of which is the tuple), and the interpretations of atoms are terms that include the variables of the adjoints, if any. As before, $D_{3s} < D$, and so we have derivations like this:



Example iG_1**.** This example is inspired by the slightly nonstandard grammar in Keenan and Stabler (2003,§2.2), in which the subject and object take their predicate as argument. The functions NOM, ACC and SELF are explicitly defined in Keenan and Stabler (2003), and can be the same here. Since the fronting of a constituent typically does not affect truth conditions, the truth conditions are given by the v phrase, and the other fronted constituents are interpreted as the identity on

the semantic domain, represented by 1:

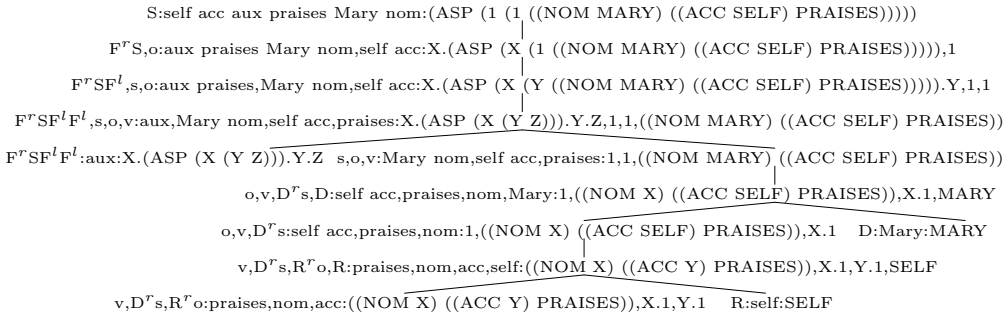
$$\begin{pmatrix} \mathrm{F^rSF^lF^l} \\ \text{aux} \\ \text{X.(ASP (X (Y Z)).Y.Z} \end{pmatrix} \begin{pmatrix} \text{v} & & \mathrm{D^rs} & \mathrm{R^ro} \\ \text{praises} & & \text{nom} & \text{acc} \\ \text{(NOM Y) ((ACC X) PRAISES)} & & \textit{Y}.1 & \textit{X}.1 \end{pmatrix}$$

$$\begin{pmatrix} \text{D} \\ \text{Mary} \\ \text{MARY} \end{pmatrix} \begin{pmatrix} \text{D} \\ \text{John} \\ \text{JOHN} \end{pmatrix} \begin{pmatrix} \text{R} \\ \text{self} \\ \text{SELF} \end{pmatrix}$$

We extend the identity partial order with these

$$\mathrm{D} < \mathrm{R} \qquad \mathrm{o} < \mathrm{F} \qquad \mathrm{s} < \mathrm{F} \qquad \mathrm{v} < \mathrm{F}.$$

This yields derivations like the following, together with the 5 other truth-functionally equivalent aux-2 orders of these same constituents:



7. First comparisons and alternatives

7.1. Formal minimalist grammars (MGs)

The mTPGs defined here allow derivations very similar to MG derivations, especially in the prefix, feature-driven normal form defined above. We could pull these two formalisms even closer by adding further constraints on the lexicons and grammatical operations, like these:

a. We could require that at most one coordinate in a lexical element is phonetically non-empty.

b. We could require that no lexical item have two unsaturated types.

c. We could require that the derivations be 'complement-first' in the sense that a right adjoint in a type cannot be contracted if it has a left adjoint. Then 'complements' – elements selected on the right – must be attached before 'specifiers' on the left.

But as our examples illustrate, for some constructions in human languages, mTPGs seem simpler and more natural than MGs because they allow more than one unsaturated component in an expression, and because they allow more than one component (more than one 'link') to be phonetically non-empty. Versions of Chomskian syntax with these properties have not been carefully explored. We have not specified a mapping from mTPG derivations to Chomskian X-bar like derivation structures with traces, but, especially from a prefix, complement-first, feature-driven normal form, this should not be much more difficult than for MGs.

7.2. Broader perspectives

Lambek (1958) differs with Bar-Hillel (1953) over whether the binary operations in grammar should be associative. As noted by Moortgat, Steedman and many others, a certain degree of associativity provides the structural flexibility that seems to be necessary for reasonable treatments of right-node raising and certain other constructions. The particular characteristics of MGs make it difficult to see any reasonable approach to combinations of unsaturated constituents as we would like to be able to have in even simple coordination and adjunction. A careful exploration of approaches to coordination in mTPG-like systems is beyond the scope of this paper, but Lambek's work on pregroup grammars shows how coordination could be handled if we allow non-rl types and relax the saturation requirement slightly.

In this vein, mTPGs allow an approach to Chomskian-style movement that may be worth some attention. Suppose a language learner was competent with a language like G_0 except for the wh-movement constructions, and knew some non-human nouns (e.g. *dog, cat, class, group,...*) and several prepositions which could be used with the verb *spoke* (e.g. *to, with, for*). Studies of child language acquisition of wh-questions have shown that children notice and sometimes even use the wh-words before they are capable of placing them properly. So a child would typically learn various relative pronouns for human and non-human referents (*who, which*) one at a time, but there is evidence that the realization of how these elements should be placed in the sentence happens at once for all of them (e.g. van Kampen, pp.74-75). We might similarly expect that once the child learns how to 'pied pipe' phrases built with one preposition, phrases built with other prepositions would be mastered at the same time:

the girl who we spoke to laughs	the group which we spoke to laughs
the group to which we spoke laughs	the girl to who we spoke laughs
the group with which we spoke laughs	the girl with who we spoke laughs

The extension of G_0 suggested above would not predict that the different relative pronouns would be mastered together, or that the different pied piped phrases would be mastered together, but there is a different and simpler extension which would make this prediction. If we had recognized several relative pronouns,

$$\begin{pmatrix} \mathrm{D}_{\mathrm{wh}} \\ \mathrm{who} \end{pmatrix} \begin{pmatrix} \mathrm{D}_{\mathrm{wh}} \\ \mathrm{which} \end{pmatrix},$$

they can all be properly enabled in the syntax at once by the addition of a single element which, when combined with these, yields categories like the one we proposed in our earlier treatment, namely:

$$\begin{pmatrix} \mathrm{D} & \mathrm{wD}^{\mathrm{l}}_{\mathrm{wh}} \\ \epsilon & \epsilon \end{pmatrix}.$$

Clearly, the combination of this new element with *who* will yield exactly the expression we proposed for the lexicon in §3.1, but now we will also get similar lexical entries for all the relative pronouns. Similarly, if we have several prepositions

$$\begin{pmatrix} \mathrm{PD}^{\mathrm{l}} \\ \mathrm{to} \end{pmatrix} \begin{pmatrix} \mathrm{PD}^{\mathrm{l}} \\ \mathrm{with} \end{pmatrix} \begin{pmatrix} \mathrm{PD}^{\mathrm{l}} \\ \mathrm{for} \end{pmatrix},$$

we can add a single new element to allow all these phrases to 'pied pipe' in exactly the way we saw in the derivation displayed in §3.1, namely

$$\begin{pmatrix} \mathrm{DP}^{\mathrm{r}}\mathrm{ww}^{\mathrm{l}} & \mathrm{PD}^{\mathrm{l}} \\ \epsilon & \epsilon \end{pmatrix}.$$

Notice that this last element is not rl, but a slight relaxation of the mTPG constraints, a relaxation already motivated by the straightforward approach to coordination, could allow it. A wide range of possibilities of this sort remains to be explored.

8. Summary and prospects

PGs have less categorial structure than CGs – they are associative – but still define any CFL. Adding non-logical constraints, mPGs also define any CFL, with minimal PG derivations. Extending mPGs to mTPGS,

mTPG operations are simpler than MG operations but in spite of that, mTPGs for common constructions sometimes simpler. mTPGs also keep less structure in their moving components than conventional grammars and formalizations like MGs – intuitively, the order in which chain links are attached is free, and the chains passing through any point of an analysis are amalgamated – but still mTPGs can define any language MGs can. Prospects for reasonable model theories look good (though our look was very brief). It might be necessary to extend the power of the formalism slightly (Michaelis and Kracht, 1997), but these grammars are already quite powerful.

The 'nonlogical' constraints – proper types and saturation – predict islands and play an essential role in our results and applications. It seems likely that these are not the right ones, but they are especially simple and may be a useful, computationally tractable starting point for later more sophisticated proposals. Relations to other proposals remain to be explored: the system proposed here is in some ways closer to the ideas of Lecomte and Retoré (1999) than MGs are; Moortgat, Morrill, Fadda and others are exploring logically defined locality conditions.

Acknowledgements

This work would not have been possible without the inspiring achievements and generous encouragement of Joachim Lambek.

References

Bach, E.: 1983, 'Generalized categorial grammars and the English auxiliary'. In: F. Heny and B. Richards (eds.): *Linguistic Categories, Volume 2*. Dordrecht: Riedel.

Bar-Hillel, Y.: 1953, 'A quasi-arithmetical notation for syntactic description'. *Language* **29**, 47–58. Reprinted in Y. Bar-Hillel, *Language and Information: Selected Essays on their Theory and Application*. NY: Addison-Wesley, 1964.

Brody, M.: 1995, *Lexico-Logical Form: A Radically Minimalist Theory*. Cambridge, Massachusetts: MIT Press.

Brody, M.: 1998, 'Projection and phrase structure'. *Linguistic Inquiry* **29**, 367–398.

Brody, M.: 2000, 'Mirror theory: syntactic representation in perfect syntax'. *Linguistic Inquiry* **31**, 29–56.

Buszkowski, W.: 2001, 'Lambek grammars based on pregroups'. In: P. de Groote, G. Morrill, and C. Retoré (eds.): *Logical Aspects of Computational Linguistics*, Lecture Notes in Artificial Intelligence, No. 2099. NY: Springer.

Casadio, C. and J. Lambek: 2002, 'A tale of four grammars'. *Studia Logica* **71**(3), 315–329.

Chomsky, N.: 1956, 'Three models for the description of language'. *IRE Transactions on Information Theory* **IT-2**, 113–124.

Chomsky, N.: 1995, *The Minimalist Program*. Cambridge, Massachusetts: MIT Press.

Church, A.: 1940, 'A formulation of the simple theory of types'. *Journal of Symbolic Logic* **5**, 56–68.

Cornell, T. L.: 1996, 'A minimalist grammar for the copy language'. Technical report, SFB 340 Technical Report #79, University of Tübingen.

Culicover, P. W.: 1999, *Syntactic Nuts: Hard Cases, Syntactic Theory, and Language Acquisition*. NY: Oxford University Press.

Fadda, M.: 2002, 'Towards flexible pregroup grammars'. In: *Proceedings 2002 Roma Workshop*.

Fanselow, G. and A. Mahajan: 2000, 'Toward a minimalist theory of wh-expletives, wh-copying and successive cyclicity'. In: U. Lutz, G. Müller, and von Stechow (eds.): *Wh-Scope Marking*. NY: John Benjamins, pp. 195–230.

Fitzpatrick, J. M.: 2002, 'On minimalist approaches to the locality of movement'. *Linguistic Inquiry* **33**, 443–463.

Hale, K.: 1992, 'Basic word order in two 'free word order' languages'. In: D. L. Payne (ed.): *Pragmatics of Word Order Flexibility*. Philadelphia: John Benjamins, pp. 63–82.

Harkema, H.: 2001, 'A characterization of minimalist languages'. In: *Proceedings, Logical Aspects of Computational Linguistics, LACL'01*. Port-aux-Rocs, Le Croisic, France.

Horvath, J.: 2000, 'On the syntax of 'wh-scope marker' constructions: some comparative evidence'. In: U. Lutz, G. Müller, and von Stechow (eds.): *Wh-Scope Marking*. NY: John Benjamins, pp. 271–316.

Huet, G.: 1975, 'A unification procedure for the typed λ-calculus'. *Theoretical Computer Science* **1**, 27–57.

Joshi, A.: 1985, 'How much context-sensitivity is necessary for characterizing structural descriptions'. In: D. Dowty, L. Karttunen, and A. Zwicky (eds.): *Natural Language Processing: Theoretical, Computational and Psychological Perspectives*. NY: Cambridge University Press, pp. 206–250.

Kayne, R.: 1994, *The Antisymmetry of Syntax*. Cambridge, Massachusetts: MIT Press.

Keenan, E. L. and E. P. Stabler: 2003, *Bare Grammar*. Stanford, California: CSLI Publications.

Kobele, G. M.: 2002, 'Formalizing Mirror theory'. *Grammars* **5**, 177–221.

Koopman, H. and A. Szabolcsi: 2000, *Verbal Complexes*. Cambridge, Massachusetts: MIT Press.

Kracht, M.: 2002, 'Referent systems and relational grammar'. *Journal of Logic, Language and Information* **11**, 251–286.

Lambek, J.: 1958, 'The mathematics of sentence structure'. *American Mathematical Monthly* **65**, 154–170.

Lambek, J.: 1999, 'Type grammars revisited'. In: A. Lecomte, F. Lamarche, and G. Perrier (eds.): *Logical Aspects of Computational Linguistics*, Lecture Notes in Artificial Intelligence, No. 1582. NY: Springer, pp. 1–27.

Lambek, J.: 2000, 'Pregroups: a new algebraic approach to sentence structure'. In: C. Martin-Vide and G. Păun (eds.): *Recent Topics in Mathematical and Computational Linguistics*. Bucharest: Editura Academici Române.

Lambek, J.: 2001, 'Type grammars as pregroups'. *Grammars* **4**, 21–35.

Lasnik, H.: 1995, 'Verbal morphology: *Syntactic Structures* meets the minimalist program'. In: H. Campos and P. Kempchinsky (eds.): *Evolution and Revolution in Linguistic Theory: Essays in Honor of Carlos Otero*. Baltimore: Georgetown University Press, pp. 251–275. Reprinted in H. Lasnik, editor, *Minimalist Analysis*, NY: Blackwell, 1999.

Lecomte, A. and C. Retoré: 1999, 'Towards a minimal logic for minimalist grammars'. In: *Proceedings, Formal Grammar'99*. Utrecht.

Mahajan, A.: 2000, 'Eliminating head movement'. In: *The 23rd Generative Linguistics in the Old World Colloquium, GLOW '2000, Newsletter #44*. pp. 44–45.

McDaniel, D., B. Chiu, and T. L. Maxfield: 1995, 'Parameters for wh-movement types: evidence from child English'. *Natural Language and Linguistic Theory* **13**, 709–753.

Merchant, J.: 2003, 'Subject-auxiliary inversion in comparatives and PF output constraints'. In: K. Schwabe and S. Winkler (eds.): *The Interfaces: Deriving and Interpreting Omitted Structures*. Philadelphia: John Benjamins.

Michaelis, J.: 1998, 'Derivational minimalism is mildly context-sensitive'. In: *Proceedings, Logical Aspects of Computational Linguistics, LACL'98*. NY, Springer.

Michaelis, J. and M. Kracht: 1997, 'Semilinearity as a syntactic invariant'. In: C. Retoré (ed.): *Logical Aspects of Computational Linguistics*. NY, pp. 37–40, Springer-Verlag (Lecture Notes in Computer Science 1328).

Miller, D. A.: 1998, *λ-Prolog: An Introduction to the Language and its Logic*. Pennsylvania State University: forthcoming.

Moortgat, M.: 1996, 'Categorial type logics'. In: J. van Benthem and A. ter Meulen (eds.): *Handbook of Logic and Language*. Amsterdam: Elsevier.

Moortgat, M.: 2003, 'Categorial grammar and formal semantics'. In: *Encyclopedia of Cognitive Science*. MacMillan: NY. Preliminary (long) version. Publication forthcoming.

Morrill, G. V.: 2002, 'Towards generalized discontinuity'. In: *Formal Grammar 2002, Proceedings of the Conference*. Trento.

Munro, P.: 1989, 'Postposition incorporation in Pima'. *Southwest Journal of Linguistics* **9**, 108–127.

Nadathur, G. and D. A. Miller: 1998, 'Higher order logic programming'. In: D. M. Gabbay, C. J. Hogger, and J. A. Robinson (eds.): *Handbook of Logics for Artificial Intelligence and Logic Programming*, Vol. 5. Oxford: Clarendon, pp. 499–590.

Oehrle, R. T.: 2002, 'A multi-modal perspective on Lambek's pregroups'. Lecture notes, University of Edinburgh.

Pentus, M.: 1993, 'Lambek grammars are context free'. In: *Eighth Annual IEEE Symposium on Logic in Computer Science*. pp. 429–433.

Rizzi, L.: 1990, *Relativized Minimality*. Cambridge, Massachusetts: MIT Press.

Roberts, I. G.: 1993, *Verbs and Diachronic Syntax*. Boston: Kluwer.

Ross, J. R.: 1967, 'Constraints on Variables in Syntax'. Ph.D. thesis, Massachusetts Institute of Technology.

Seki, H., T. Matsumura, M. Fujii, and T. Kasami: 1991, 'On multiple context-free grammars'. *Theoretical Computer Science* **88**, 191–229.

Smith, M.: 2002, 'Partial Agreement in Pima'. UCLA M.A. thesis.

Stabler, E. P.: 1994, 'The finite connectivity of linguistic structure'. In: C. Clifton, L. Frazier, and K. Rayner (eds.): *Perspectives on Sentence Processing*. Hillsdale, New Jersey: Lawrence Erlbaum, pp. 245–266.

Stabler, E. P.: 1997, 'Derivational minimalism'. In: C. Retoré (ed.): *Logical Aspects of Computational Linguistics*. NY: Springer-Verlag (Lecture Notes in Computer Science 1328), pp. 68–95.

Stabler, E. P.: 2001, 'Recognizing head movement'. In: P. de Groote, G. Morrill, and C. Retoré (eds.): *Logical Aspects of Computational Linguistics*, Lecture Notes in Artificial Intelligence, No. 2099. NY: Springer, pp. 254–260.

Stabler, E. P. and E. L. Keenan: 2003, 'Structural similarity'. *Theoretical Computer Science* **293**, 345–363.

Thornton, R.: 1990, 'Adventures in Long-Distance Moving: The Acquisition of Complex Wh-Questions'. Ph.D. thesis, University of Connecticut.

van Kampen, J.: 1997, 'First steps in wh-movement'. Ph.D. thesis, University of Utrecht.

Vermaat, W.: 2004, 'The minimalist move operation in a deductive perspective'. *Research on Language and Computation* **2**(1), 69–85.

Vermeulen, K. and A. Visser: 1996, 'Dynamic bracketing and discourse representation'. *Notre Dame Journal of Formal Logic* **37**, 321–365.

Weir, D.: 1988, 'Characterizing mildly context-sensitive grammar formalisms'. Ph.D. thesis, University of Pennsylvania, Philadelphia.