Recognizing head movement

Edward P. Stabler

University of California, Los Angeles CA 90095-1543, USA,
http://www.linguistics.ucla.edu/people/stabler/

Abstract. Previous studies have provided logical representations and efficient recognition algorithms for a simple kind of "minimalist grammars." This paper extends these grammars with head movement ("incorporation") and affix hopping. The recognition algorithms are elaborated for these grammars, and logical perspectives are briefly considered.

Michaelis (1998) showed how the derivations of a simple kind of minimalist grammar (MG) (Stabler, 1997) correspond to derivations of exactly the same strings in a multiple context free grammar (MCFG) (Seki et al., 1991). MGs build structures by *merging* pairs of expressions, and simplifying single expressions by *moving* a subexpression in them. The basic idea behind the correspondence with MCFGs can be traced back to Pollard (1984) who noticed, in effect, that when a constituent is going to move, we should not regard its yield as included in the yield of the expression that contains it. Instead, the expression from which something will move is better regarded has having multiple yields, multiple components – the "moving" components have not yet reached their final positions. In a completed derivation, the component strings of all the categories are eventually ordered to yield a sentence. This conception behind the Michaelis result relies on the fact that for any MG, there is a fixed, finite bound on the number of categories and rules, and on the number of components that any constituent will have.

These recent results led to efficient parsing methods (Stabler, 1999; Harkema, 2000), to connections with multimodal logics (Cornell, 1999; Vermaat, 1999; Lecomte and Retoré, 1999), and to a succinct reformulation of MGs (Stabler and Keenan, 2000) in which the multiple components of expressions are explicit. The parsing methods, the logics, and the succinct reformulation were provided for MGs with only overt, phrasal movement (as explained below). This paper extends these results with two kinds of head movement, providing an efficient parsing method and some preliminary observations about bringing this work into a logical perspective like that proposed by (Lecomte and Retoré, 1999). And for illustration, in the course of the paper, we present small example grammars for (G1) subject-auxiliary inversion and (G3) affix-hopping in English, and (G2) object clitics in French.

1 Minimalist grammars with phrasal movement

We begin from the succinct presentation of MGs introduced in (Stabler and Keenan, 2000). We will define a set of linguistic expressions which are built from a nonempty set of pronounced elements Σ, a two-element set of types $T = \{::, :\}$, and a set of features F (where the sets Σ, T, F are pairwise disjoint). F is partitioned into four sets, given by a nonempty set of basic features $B \subset F$ and three functions with domain C and disjoint ranges: **base** $B = \{v, n, np, case, wh, \ldots\}$, **selectors** $S = \{{=}f |\ f \in B\}$, **licensees** $M = \{-f |\ f \in B\}$, **licensors** $N = \{+f |\ f \in B\}$. So the set of features $F = B \cup S \cup M \cup N$.

Let the set of **chains** $C = \Sigma^* T F^*$. We use the types $T = \{::, :\}$ to distinguish lexical chains from derived chains, where :: is the lexical type, so the set of **lexical chains** $LC = \Sigma^* :: F^*$. The set of **expressions** $E = C^+$, the set of nonempty sequences of chains. When an expression has more than one chain, we will separate the chains with commas to enhance readability. A **lexicon** Lex is a finite set of lexical chains. And finally, a **minimalist grammar** G is just a lexicon, $G = Lex$.

For any minimalist grammar $G = Lex$, the **language** $L(G)$ is the closure of Lex under the fixed set of structure building functions $\mathcal{F} = \{merge, move\}$, where these functions are defined below. And for any $\mathtt{f} \in B$, the strings of category $\mathtt{f}$, $S_{\mathtt{f}}(G) = \{s |\ s \cdot \mathtt{f} \in L(G) \text{ for some } \cdot \in T\}$.

$merge : (E \times E) \to E$ is the union of the following 3 functions, for $s, t \in \Sigma^*$, for $\cdot \in \{:, ::\}$, for $f \in base$, $\gamma \in F^*$, $\delta \in F^+$, and for chains $\alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l$ $(0 \leq k, l)$

$$\frac{s :: {=}f\gamma \qquad t \cdot f, \alpha_1, \ldots, \alpha_k}{st : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1}$$

$$\frac{s : {=}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t \cdot f, \iota_1, \ldots, \iota_l}{ts : \gamma, \alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l}\ \text{r2}$$

$$\frac{s \cdot {=}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t \cdot f\delta, \iota_1, \ldots, \iota_l}{s : \gamma, \alpha_1, \ldots, \alpha_k, t : \delta, \iota_1, \ldots, \iota_l}\ \text{r3}$$

Here st is the concatenation of strings s, t. And note that since the domains of r1, r2, and r3 are disjoint, their union is a function.

$move : E \to E$ is the union of the following 2 functions, for $s, t \in \Sigma^*$, $f \in base$, $\gamma \in F^*$, $\delta \in F^+$, and for chains $\alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l$ $(0 \leq k, l)$ satisfying the following condition: (SMC) none of $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k$ has $-f$ as its first feature.

$$\frac{s : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f, \alpha_{i+1}, \ldots, \alpha_k}{ts : \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k}\ \text{m1}$$

$$\frac{s : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f\delta, \alpha_{i+1}, \ldots, \alpha_k}{s : \gamma, \alpha_1, \ldots, \alpha_{i-1}, t : \delta, \alpha_{i+1}, \ldots, \alpha_k}\ \text{m2}$$

Notice that the domains of m1 and m2 are disjoint, so their union is a function. The (SMC) restriction on the domain of *move* is a simple version of the "shortest move condition" (Chomsky, 1995).

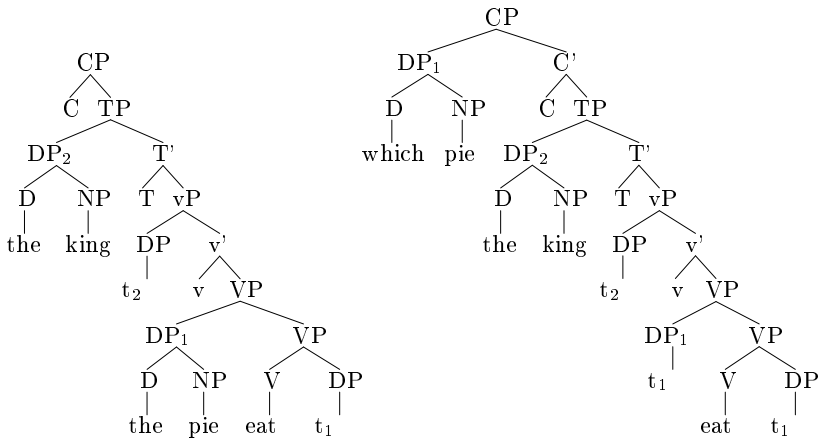
1.1 G0: wh-movement in a simple SOV language

A simple approach to wh-movement allows us to derive simple sentences and wh-questions like the following, in an artificial Subject-Object-Verb language with no verbal inflections:

(1) the king the pie eat

(2) which pie the king eat

Linguists have proposed that not only is the question formed by moving the wh determiner phrase (DP) [which king] from object position to the front, but in all clauses the pronounced DPs move to case positions, where transitive verbs assign case to their objects ("Burzio's generalization"). So then the clauses above get depictions rather like this, indicating movements by leaving coindexed "traces" (t) behind:

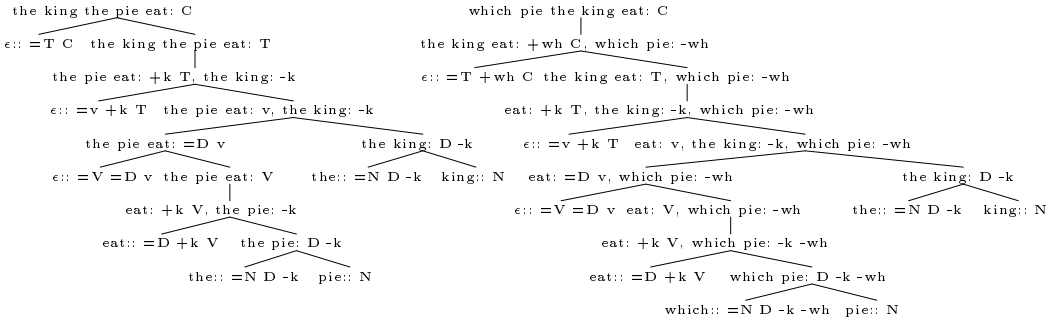


As indicated by coindexing, in the tree on the left, there are two movements, while the tree on the right has three movements because [which pie] moves twice: once to a case position, and then to the front, wh-question position. The sequences of coindexed constituents are sometimes called "chains."

These expressions can be defined by an MG with the following 10 lexical items (writing ϵ for the empty string, and using `k` for the abstract "case" feature):

ϵ:: =T C	ϵ:: =T +wh C
ϵ:: =v +k T	ϵ:: =V =D v
eat:: =D +k V	laugh:: V
the:: =N D -k	which:: =N D -k -wh
king:: N	pie:: N

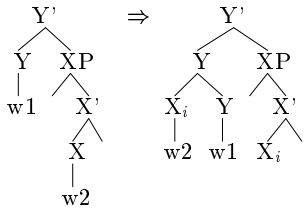
With this grammar, we can derive strings of category C as follows, where in these trees the leaves are lexical items, a node with two daughters represents the result of *merge*, and a node with one daughter represents the result of a *move*.



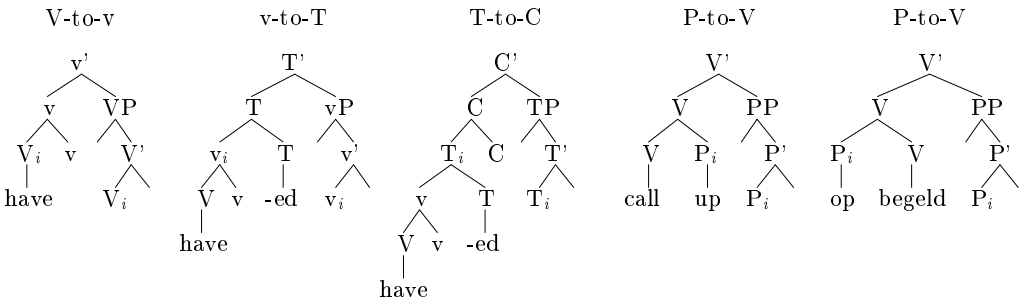
Since *merge* is binary and *move* is unary, it is easy to see that the tree on the left has two movements, while the one on the right has three. A full discussion of the correspondence between the linguists' conception of derivations and the one formally defined here is beyond the scope of this paper, but the correspondence is very close (Stabler, 1999).

2 Incorporation

Many linguists believe that in addition to phrasal movement of the sort discussed above, there is "head movement", which moves not the whole phrase but just the "head". In the simplest, "canonical" examples, a head X of a phrase XP moves to adjoin to the left or right of the head Y that selects XP. Left-adjoining X to Y is often depicted this way:



For example, questions with inversion of subject and inflected verb may be formed by moving the T head to C (sometimes called T-to-C or I-to-C movement); verbs may get their inflections by V-to-T movement; particles may get associated with verbs by P-to-V movement; objects may incorporate into the verb with N-to-V movement, and there may also be v-to-V movement.



As indicated by these examples of v-to-T and T-to-C movement, heads can be complex. And notice that the P-to-V movement is right-adjoining in the English [call up] but left-adjoining in the Dutch [opgebeld] (Koopman 1993, 1994). Similarly (though not shown here) when a verb incorporates a noun, it is usually attached on the left, but sometimes on the right (Baker, 1996, 32).

The MGs defined above can be extended to allow these sorts of movements. Since they involve the configuration of selection, we follow the suggestion in Stabler (1997) that they be regarded as part of the *merge* operation. Remembering the essential insight from Pollard and Michaelis, mentioned on the first page, the key thing is to keep the phonetic contents of any movable head in a separate component. A head X not movable after its phrase XP has been merged, so we only need to distinguish the head components of phrases until they have been merged. So rather than expressions of the form:

$$\texttt{s}_1 \cdot \texttt{Features}_1,\ \texttt{s}_2 \cdot \texttt{Features}_2,\ldots,\texttt{s}_k \cdot \texttt{Features}_k,$$

we will use expressions in which the string part $\texttt{s}_1$ of the first chain is split into three (possibly empty) pieces `s(pecifier), h(ead), c(omplement)`:

$$\texttt{s,h,c} \cdot \texttt{Features}_1,\ \texttt{s}_2 \cdot \texttt{Features}_2,\ldots,\texttt{s}_k \cdot \texttt{Features}_k.$$

So **lexical chains** now have a triple of strings, but only the head can be non-empty: $LC = \epsilon, \Sigma^*, \epsilon :: F^*$. As before, a **lexicon** is a finite set of lexical chains.

Head movement will be triggered by a specialization of the selecting feature. The feature `=>V` will indicate that the head of the selected VP is to be adjoined on the left; and `V<=` will indicate that the head of the selected VP is to be adjoined on the right. The former set of features is thus extended by adding these two new functions on the base categories B: **right-incorporators** $R = \{f\texttt{<=} \mid f \in B\}$, and **left-incorporators** $L = \{\texttt{=>}f \mid f \in B\}$. So now the set of syntactic features $F = B \cup S \cup M \cup N \cup R \cup L$. The new work of placing heads properly is done by the *merge* function, so the earlier functions r1 and r3 each break into 3 cases. Define *merge* as the union of the following 7 functions:

$$\frac{\epsilon, s, \epsilon :: \texttt{=}f\gamma \qquad t_s, t_h, t_c \cdot f, \alpha_1, \ldots, \alpha_k}{\epsilon, s, t_s t_h t_c : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1'}$$

$$\frac{\epsilon, s, \epsilon :: f\texttt{<=}\gamma \qquad t_s, t_h, t_c \cdot f, \alpha_1, \ldots, \alpha_k}{\epsilon, st_h, t_s t_c : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1right}$$

$$\frac{\epsilon, s, \epsilon :: \texttt{=>}f\gamma \qquad t_s, t_h, t_c \cdot f, \alpha_1, \ldots, \alpha_k}{\epsilon, t_h s, t_s t_c : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1left}$$

$$\frac{s_s, s_h, s_c : \texttt{=}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f, \iota_1, \ldots, \iota_l}{t_s t_h t_c s_s, s_h, s_c : \gamma, \alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l}\ \text{r2'}$$

$$\frac{s_s, s_h, s_c \cdot \texttt{=}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f\delta, \iota_1, \ldots, \iota_l}{s_s, s_h, s_c : \gamma, \alpha_1, \ldots, \alpha_k, t_s t_h t_c : \delta, \iota_1, \ldots, \iota_l}\ \text{r3'}$$

$$\frac{s_s, s_h, s_c :: f\texttt{<=}\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f\delta, \iota_1, \ldots, \iota_l}{s_s, s_h t_h, s_c : \gamma, \alpha_1, \ldots, \alpha_k, t_s t_c : \delta, \iota_1, \ldots, \iota_l}\ \text{r3right}$$

$$\frac{s_s, s_h, s_c :: \texttt{=>}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f\delta, \iota_1, \ldots, \iota_l}{s_s, t_h s_h, s_c : \gamma, \alpha_1, \ldots, \alpha_k, t_s t_c : \delta, \iota_1, \ldots, \iota_l}\ \text{r3left}$$

And *move* changes only trivially. It is the union of the following functions:

$$\frac{s_s, s_h, s_c : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f, \alpha_{i+1}, \ldots, \alpha_k}{t s_s, s_h, s_c : \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k}\ \text{m1'}$$

$$\frac{s_s, s_h, s_c : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f\delta, \alpha_{i+1}, \ldots, \alpha_k}{s_s, s_h, s_c : \gamma, \alpha_1, \ldots, \alpha_{i-1}, t : \delta, \alpha_{i+1}, \ldots, \alpha_k}\ \text{m2'}$$

As before, for any grammar $G = Lex$, the language $L(G)$ is the closure of Lex under the fixed set of structure building functions $\mathcal{F} = \{merge, move\}$. And for any $\texttt{f} \in B$, the strings of category $\texttt{f}$, $S_{\texttt{f}}(G) = \{s_s s_h s_c |\ s_s, s_h, s_c \cdot \texttt{f} \in L(G) \text{ for some } \cdot \in T\}$.

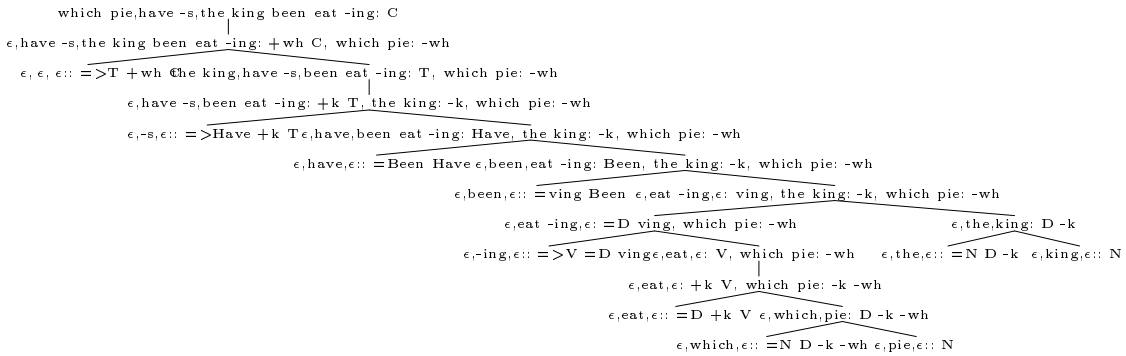
2.1 G1: Subject-auxiliary inversion in English

Introductions to transformational syntax like (Fromkin, 2000, §5) often present a simplified account of English auxiliaries and question formation that can now be represented with a lexicon like the following (writing $\texttt{s}_h\texttt{::}\ \texttt{Fs}$ for each $(\texttt{s}_s\texttt{,s}_h\texttt{,s}_c\texttt{::}\ \texttt{Fs}) \in Lex$, since $\texttt{s}_s$ and $\texttt{s}_c$ are always empty in the lexicon):[1]

ϵ:: =T C	ϵ:: =>T C	ϵ:: =>T +wh C	
-s:: =>Modal +k T	-s:: =>Have +k T	-s:: =>Be +k T	-s:: =v +k T
will:: =Have Modal	will:: =Be Modal	will:: =v Modal	
have:: =Been Have	have:: =v_{en} Have		
be:: =v_{ing} Be	been:: =v_{ing} Been		
ϵ:: =>V =D v	-en:: =>V =D v_{en}	-ing:: =>V =D v_{ing}	
eat:: =D +k V	laugh:: V		
the:: =N D -k	which:: =N D -k -wh		
king:: N	pie:: N		

[1] I follow the linguistic convention of punctuating a string like `-s` to indicate that it is an affix. This dash that occurs next to a string should not be confused with the dash that occurs next to syntactic features like `-wh`.

With this grammar we have derivations like the following



The behavior of this grammar is English-like on a range of constructions:

(3) will -s the king laugh

(4) the king be -s laugh -ing

(5) which king have -s eat -en the pie

(6) the king will -s have been eat -ing the pie

We also derive

(7) -s the king laugh

which will be discussed in §4.

2.2 G2: A simple approach to French clitics

(Sportiche, 1998) reviews some of the difficulties in providing an account of French clitics. Following Kayne (1989) and others, he points out that they act like heads attached to a verb when, for example, they move with the verb in "complex inversion" (just as the verb and inflection move together in English questions, in the previous section):

(8) Jean [l'aurait]$_i$-il t$_i$ connu?
John him-would-have-he known
'Would John have known him?'

But clitics are also related to to argument positions, and these need not be adjacent to the verb:

(9) Il lui$_i$ donnera le chapeau t$_i$
He to-him will-give the hat
'He will give him the hat'

A clitic can be associated with the argument position of a verb other than the one it is attached to:

(10) Jean la_i veut manger t_i
John it wants to-eat

'John wants to eat it'

And finally, a verb can be associated with multiple clitics:

(11) Il le_j lui_i donnera t_j t_i
he it to-him will-give

'he will give it to him'

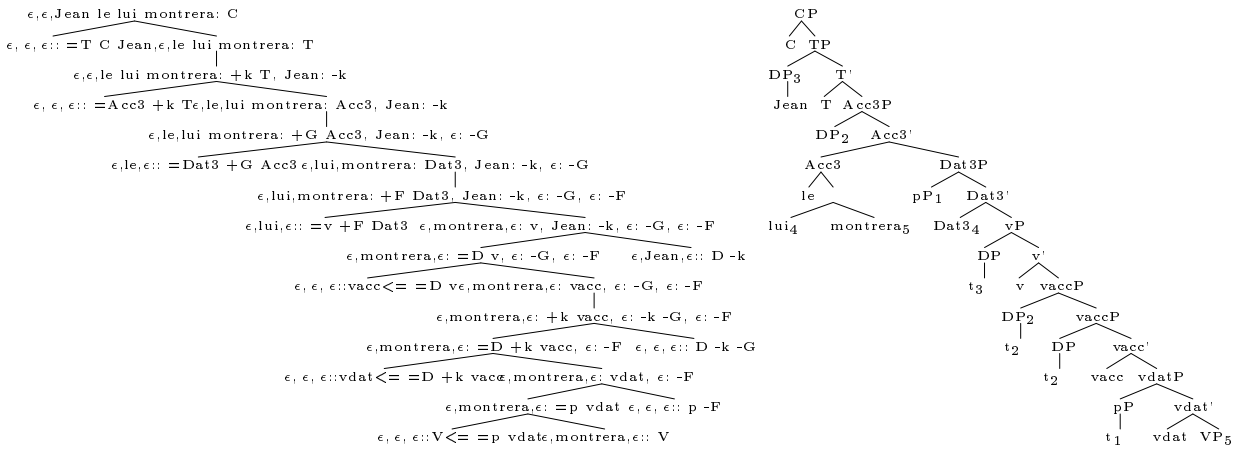
However, there is a limit on the number of clitics that can be associated with a verb in French. They are limited to at most one from each of the following sets, and at most one from the third and fifth sets together (Sportiche, 1998, 247):

Nom	Neg	Refl12	Acc3	Dat3	Loc	Gen
{il}	{ne}	{me,te,se,nous}	{le,la,les}	{lui,leur}	{y}	{en}

One way to get the proper order of these clitics that is considered by (Sportiche, 1995, 266ff) involves assuming that over V there is a projection for each of these sets. Then a Dat3 head can select V to form [lui V] by (right adjoining) head movement, which in turn can be selected by Acc3 to form [le lui V], and so on. And the association with argument positions can be accomplished by moving not the clitics themselves, but phrases ("operators," phonetically empty), so that the argument positions are properly "bound" by the clitics. If the binding of arguments by clitics in the 3rd and 5th set is accomplished by the same movement feature +F, then the (SMC) will prevent both from occurring at once, as desired. This approach is captured by a grammar like the following:

ϵ::=T C				
ϵ::=Refl12 +k T	ϵ::=Acc3 +k T	ϵ::=Dat3 +k T	ϵ::=v +k T	
se::=Acc3 +F Refl12	se::=Dat3 +F Refl12	se::=v +F Refl12		
le::=Dat3 +G Acc3	le::=v +G Acc3			
lui::=v +F Dat3				
ϵ::vacc<= =D v	ϵ::vdat<= =D +k vacc	ϵ::V<= =p vdat	montrera::V	
ϵ::P<= p	a::=D +k P	ϵ::p -F		
Jean::D -k	Marie::D -k	le::=N D -k	ϵ::D -k -F	ϵ::D -k -G
roi::N	livre::N			

With this grammar, we have derivations like the following, with the more conventional transformational depiction on the right. Notice that an element has moved from the lower P specifier of the vdatP argument of the verb to the Dat3P clitic phrase:



The reader can check that with this grammar we also have, as desired:

(12) Jean montrera le livre à Marie

(13) Jean se montrera à Marie

(14) *Jean se lui montrera

The proper semantics for these constructions must be left to another paper, but see (Sportiche, 1998) for an informal discussion of what is intended.

This simple syntax of clitic constructions leaves out agreement and many other important phenomena, but many of the more sophisticated recent proposals similarly mix phrasal and head movement. Agreement is carefully considered in (Sportiche, 1998), and interesting accounts of stylistic inversion and subject clitic inversion in French are provided in (Kayne and Pollock, 1999; Pollock et al., 1999; Poletto and Pollock, 1999). It appears that all these fall easily in the scope of the mechanisms proposed here.

3 Affix hopping

The grammar of §2.1 does not derive the simple tensed clause: *the king eat -s the pie*. The problem is that if we simply allow the verb *eat* to pick up this inflection by head movement to T, as the auxiliary verbs do, then we will mistakenly also derive **eat -s the king the pie*. Since Chomsky (1957), one common proposal is that when there is no auxiliary verb, the inflection can lower to the main verb. This lowering is sometimes called "affix hopping." In the present context, it is interesting to notice that once the head of unmerged phrases is distinguished for head movement, no further components are required for affix hopping.

We can formalize this idea in our grammars as follows. We introduce two new kinds of features $\texttt{<=}f$ and $f\texttt{=>}$ (for any $f \in B$), and we add the following additional cases to definition of *merge*:

$$\frac{\epsilon, s, \epsilon :: f\texttt{=>}\gamma \qquad t_s, t_h, t_c \cdot f, \alpha_1, \ldots, \alpha_k}{\epsilon, \epsilon, t_s t_h s t_c : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1hopright}$$

$$\frac{\epsilon, s, \epsilon :: \texttt{<=}f\gamma \qquad t_s, t_h, t_c \cdot f, \alpha_1, \ldots, \alpha_k}{\epsilon, \epsilon, t_s s t_h t_c : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1hopleft}$$

$$\frac{\epsilon, s, \epsilon :: f\texttt{=>}\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f\delta, \iota_1, \ldots, \iota_l}{\epsilon, \epsilon, \epsilon : \gamma, \alpha_1, \ldots, \alpha_k, t_s t_h s t_c : \delta, \iota_1, \ldots, \iota_l}\ \text{r3hopright}$$

$$\frac{\epsilon, s, \epsilon :: \texttt{<=}f\gamma, \alpha_1, \ldots, \alpha_k \qquad t_s, t_h, t_c \cdot f\delta, \iota_1, \ldots, \iota_l}{\epsilon, \epsilon, \epsilon : \gamma, \alpha_1, \ldots, \alpha_k, t_s s t_h t_c : \delta, \iota_1, \ldots, \iota_l}\ \text{r3hopleft}$$

This formulation of affix-hopping as a sort of string-inverse of head movement has the consequence that an affix can only "hop" to the head of a selected phrase, not to the head of the head selected by a selected phrase. That is, affix hopping can only take place in the configuration of selection.[2] It is now a simple matter to obtain a grammar G2 that gets simple inflected clauses.

3.1 G3: Affix hopping in English

We elaborate grammar G1 by adding a single lexical item:

```
-s:: v=> +k T
```

It is left as an exercise for the reader to verify that the set of strings of category `C` now allows main verbs to be inflected but not fronted, as desired:

(15) the king eat -s the pie

(16) *eat -s the king the pie

This kind of account of English clause structure commonly adds one more ingredient: do-support. Introductory texts sometimes propose that *do* can be attached to any stranded affix, perhaps by a process that is not part of the syntax proper. We accordingly take it up in the next section.

4 Recognition

4.1 From input to syntactic atoms

Phonological and morphological analysis of an acoustic input will commonly yield more than one possible analysis of the input to be parsed. Sometimes it

[2] (Sportiche, 1998, 382) points out that the proposal in (Chomsky, 1993) for avoiding affix hopping also has the consequence that affixes on main verbs in English can only occur in the configuration where head movement would also have been possible.

is assumed that the set of possible analyses can be represented with a regular grammar or finite state machine. We will adopt that idea here, implementing a simple kind of "morphological analysis" with a finite state transducer.

For any set S, let $S^\epsilon = (S \cup \{\epsilon\})$. Then as usual, a **finite state machine**(FSM) $A = \langle Q, \Sigma, \delta, I, F\rangle$ where Q is a finite set of states ($\neq \emptyset$); Σ_1 is a finite set of input symbols ($\neq \emptyset$); $\delta \subseteq Q \times \Sigma^\epsilon \times Q$, $I \subseteq Q$, the initial states; $F \subseteq Q$, the final states. Intuitively, a **finite transducer** is an acceptor where the transitions between states are labeled by pairs. Formally, we let the pairs come from different alphabets: $T = \langle Q, \Sigma_1, \Sigma_2, \delta, I, F\rangle$ where Q is a finite set of states ($\neq \emptyset$); Σ_1 is a finite set of input symbols ($\neq \emptyset$); Σ_2 is a finite set of output symbols ($\neq \emptyset$); $\delta \subseteq Q \times \Sigma_1^\epsilon \times \Sigma_2^\epsilon \times Q$, $I \subseteq Q$, the initial states; $F \subseteq Q$, the final states. And as usual, we assume that for any state q and any transition function δ, $\langle q, \epsilon, \epsilon, q\rangle \in \delta$.

For any transducers $T = \langle Q, \Sigma_1, \Sigma_2, \delta_1, I, F\rangle$ and $T' = \langle Q', \Sigma_1', \Sigma_2', \delta_2, I', F'\rangle$, define the **composition** $T \circ T' = \langle Q \times Q', \Sigma_1, \Sigma_2', \delta, I \times I', F \times F'\rangle$ where $\delta = \{\langle\langle q_i, q_i'\rangle, a, b, \langle q_j, q_j'\rangle\rangle |$ for some $c \in (\Sigma_2^\epsilon \cap \Sigma_1'^\epsilon)$, $\langle q_i, a, c, q_j\rangle \in \delta_1$ and $\langle q_i', c, b, q_j'\rangle \in \delta_2\}$ (Kaplan and Kay, 1994, for example). And finally, for any transducer $T = \langle Q, \Sigma_1, \Sigma_2, \delta, I, F\rangle$ let its **second projection** $2(T)$ be the FSM $A = \langle Q, \Sigma_1, \delta', I, F\rangle$, where $\delta' = \{\langle q_i, a, q_j\rangle |$ for some $b \in \Sigma_2^\epsilon$, $\langle q_i, a, b, q_j\rangle \in \delta\}$.

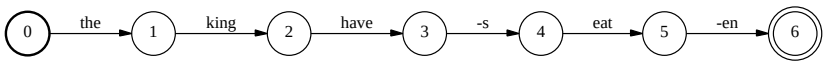
Now for any input $s \in V^*$ where $s = w_1 w_2 \ldots w_n$ for some $n \geq 0$, let $string(s)$ be the transducer $\langle\{0, 1, \ldots, n\}, \Sigma, \Sigma, \delta_0, \{0\}, \{n\}\rangle$, where $\delta = \{\langle i - 1, w_i, w_i, i\rangle | \ 0 \leq i\}$. A (finite state) **morphology** is a transducer $M = \langle Q, V, \Sigma, \delta, I, F\rangle$ such that for any $s \in V^*$, $2(string(s) \circ M)$ represents the sequences of syntactic atoms to be parsed with a grammar whose vocabulary is Σ. For any morphology M, let the function $input_M$ from V^* to Σ^* be such that for any $s \in V^*$, $input(s) = 2(string(s) \circ M)$.

4.2 M0: A simple morphology for G3

Let $M0$ be the 4-state transducer $\langle\{A, B, C, D\}, V, \Sigma, \delta, \{A\}, \{A\}\rangle$ where δ is the set containing the following 4-tuples:

⟨A,the,the,A⟩	⟨A,has,have,B⟩	⟨A,eaten,eat,C⟩	⟨A,eating,eat,D⟩
⟨A,king,king,A⟩	⟨A,is,be,B⟩	⟨A,laughed,laugh,C⟩	⟨A,laughing,laugh,D⟩
⟨A,pie,pie,A⟩	⟨A,eats,eat,B⟩	⟨C,ϵ,-en,A⟩	⟨D,ϵ,-ing,A⟩
⟨A,which,which,A⟩	⟨A,laughs,laugh,B⟩		
⟨A,eat,eat,A⟩	⟨A,will,will,B⟩		
⟨A,laugh,laugh,A⟩	⟨B,ϵ,-s,A⟩		
⟨A,does,-s,A⟩			

With this morphology, $input_{M0}$(`the king has eaten`) is the FSM depicted below, a machine that accepts only `the king have -s eat -en`:



Notice that the last triple in the left column above provides a simple kind of do-support, so that $input_{M0}$(`what does the king eat`) is the FSM that accepts

only: `what -s the king eat`. This is like example (7) from §2.1, and we see here the the beginning of one of the traditional accounts of this construction.

These values of $input_{M0}$ have just one successful path, but this is not required. Obviously, more complex morphologies (and phonologies) can be represented by FSMs (Ellison, 1994; Eisner, 1997).

4.3 From syntactic atoms to phrase structures

The input to syntactic analysis is a finite state machine $input(s) = \langle Q, V, \Sigma, \delta, I, F\rangle$. The analysis is computed by closing the set δ under the *move* and *merge*, where these functions are given exactly as above, except that each string t is represented by a pair of states that is connected by a path in $2(input(s))$ that is labeled by t. So for example, the definition of *move* given in §2 above includes the case r1', which, now representing strings as pairs of states, is the function which for any states $w, x, y_0, y, z_0, z_1, z_2, z \in Q$, any any $\cdot \in \{:, ::\}$, any $f \in base$, $\gamma \in F^*$, and any chains $\alpha_1, \ldots, \alpha_k$ $(0 \leq k, l)$ maps arguments to values as follows:

$$\frac{(x,x),(y_0,y),(w,w) :: \texttt{=}f\gamma \qquad (z_0,z_1),(z_1,z_2),(z_2,z)\cdot f, \alpha_1, \ldots, \alpha_k}{(x,x),(y_0,y),(z_0,z) : \gamma, \alpha_1, \ldots, \alpha_k}\ \text{r1'},$$

Treating all the other cases similarly, this closure can be computed with the semi-naive closure algorithm implemented by "chart parsers" like the one in (Shieber et al., 1993).[3]

Harkema (2000) shows that the complexity of this recognition algorithm, when restricted to phrasal movement, is $\mathcal{O}(n^{4k+4})$ where $n = |Q|$ and k is the number of different licensee features that occur in lexical items of the grammar. Adding two new components to the head component of each chain increases the maximum number of items just as adding 2 new licensees would, and the recognition procedure is otherwise unchanged in its essentials, so the bound remains polynomial.

5 Logical perspectives

One of the appeals of type-logical approaches to language is their connection to semantics. The structure building function *merge* corresponds to function application, and this part of the grammar captures basic predicate-argument relations. Phrasal movement has rather complex and subtle effects on scope relations, but head movement does not. Head movement appears to be a semantically empty rearrangement of phonetic material, something that can be handled by "structural rules" (Vermaat, 1999; Moortgat, 1996; Morrill, 1994), or in the labels of a labeled type-logical deduction. The approaches of (Lecomte and Retoré, 1999; Lecomte, 1999), based on a partially commutative linear logic, come especially close to those presented here, because string manipulation is relegated to the labels, and the effect of (SMC) is captured transparently with a restriction on

[3] An implementation is available from the author.

derivations. Let's briefly consider how that approach might be elaborated to capture deductions like those explored here.

The basic idea of Lecomte and Retoré is to keep the logical analysis simple and close to the linguists' conception: all the deductive steps are elimination rules; these steps must be taken in a prescribed order; and the string label calculation is separately stated. In this framework, it appears that two modes of application may suffice: one that is order-sensitive, and one that is not. In the grammars proposed here, we still have only two elimination steps: move and merge, but the string labels are now tuples of strings, and the way the string labels are calculated depends on the particular sub-case of merge or move being applied. The prospects for developing a rigorous logical perspective along these lines look promising, and the hope is that by factoring the theory into these components, the basic type logic can remain much simpler than would otherwise be possible.

6 Conclusions and assessments

The "transformational" tradition in syntax encompasses many diverse theoretical proposals, and rather wide-ranging empirical research. Formal tools can provide useful insights into this work, making the basic ideas more accessible to other frameworks. A great effort and years of critical assessment have gone into theories that involve both head- and phrasal-movement, and this paper makes just a beginning toward a clearer and simpler formal conception of this work, following the pioneering studies of (Michaelis, 1998; Harkema, 2000). Some researchers think that the role of head movement should be reduced or eliminated (Koopman and Szabolcsi, 2000; Mahajan, 2000), but very many linguists are unconvinced, and so the mechanisms explored in this paper remain of interest. Many other proposals in the recent "minimalist" tradition remain to be carefully explored, including the various proposals about "asymmetric" feature checking (Chomsky, 1995; Collins, 1997, for example), and proposals about trans-derivational "economy" or "optimality" (Pesetsky, 1989; Sportiche, 1998; Chomsky, 1995; Chomsky, 1999; Bresnan, 2000; Legendre, 2000).

It is clear that all of the MG extensions proposed here can be implemented in MCFGs. That is, the Michaelis (1998) result can be extended to all of them. Recently, Harkema (2001) has provided the converse to that result: all languages defined by MCFGs can be defined (and in a very similar way) by MGs. These results then have the perhaps surprising consequence that none of the elaborations proposed here change the definable languages. The languages defined by MCFGs, MGs, MGs with head movement, and by MGs with head movement and affix hopping, are all exactly the same. Obviously, then, the linguistic motivation for these elaborations then cannot come from differences in the definable languages. What does it come from? A full discussion is beyond the scope of this paper, but clearly it has something to do with the simplicity and elegance of the definition.

References

Mark Baker. 1996. *The Polysynthesis Parameter*. Oxford University Press, NY.

Joan Bresnan. 2000. Optimal syntax. In *Optimality Theory: Phonology, Syntax and Acquisition*. Oxford University Press, Oxford.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague.

Noam Chomsky. 1993. A minimalist program for linguistic theory. In Kenneth Hale and Samuel Jay Keyser, editors, *The View from Building 20*. MIT Press, Cambridge, Massachusetts.

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

Noam Chomsky. 1999. Derivation by phase. MIT. Forthcoming.

Chris Collins. 1997. *Local Economy*. MIT Press, Cambridge, Massachusetts.

Thomas L. Cornell. 1999. Lambek calculus for transformational grammar. In *Proceedings of the Workshop on Resource Logics and Minimalist Grammars, ESSLLI'99*, Utrecht.

Jason Eisner. 1997. Efficient generation in Primitive Optimality Theory. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

Mark T. Ellison. 1994. Phonological derivation in optimality theory. In *Procs. 15th Int. Conf. on Computational Linguistics*, pages 1007–1013. (Also available at the Edinburgh Computational Phonology Archive).

Victoria Fromkin, editor. 2000. *Linguistics: An Introduction to Linguistic Theory*. Basil Blackwell, Oxford.

Henk Harkema. 2000. A recognizer for minimalist grammars. In *Sixth International Workshop on Parsing Technologies, IWPT'2000*.

Henk Harkema. 2001. MCFG $\equiv$ MG. UCLA manuscript, forthcoming.

Ronald Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–378.

Richard Kayne and Jean-Yves Pollock. 1999. New thoughts on stylistic inversion. Manuscript, New York University and CNRS, Lyon.

Richard Kayne. 1989. Null subjects and clitic climbing. In O. Jaeggli and K. Safir, editors, *The Null Subject Parameter*. Kluwer, Dordrecht.

Hilda Koopman and Anna Szabolcsi. 2000. *Verbal Complexes*. MIT Press, Cambridge, Massachusetts.

Hilda Koopman. 1993. The structure of Dutch PPs. UCLA manuscript.

Hilda Koopman. 1994. Licensing heads. In David Lightfoot and Norbert Hornstein, editors, *Verb Movement*, pages 261–296. Cambridge University Press, NY.

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

Alain Lecomte. 1999. Rebuilding MP on a logical ground. In *Workshop on Resource Logics and Minimalist Grammars, ESSLLI'99*.

Geraldine Legendre. 2000. An introduction to optimality theory in syntax. In Geraldine Legendre, Jane Grimshaw, and Sten Vikner, editors, *Optimality-theoretic Syntax*. MIT Press, Cambridge, Massachusetts.

Anoop Mahajan. 2000. Eliminating head movement. In *The 23rd Generative Linguistics in the Old World Colloquium, GLOW '2000*.

Jens Michaelis. 1998. Derivational minimalism is mildly context-sensitive. In *Proceedings, Logical Aspects of Computational Linguistics, LACL'98*, Grenoble. Available at `http://www.ling.uni-potsdam.de/~michael/papers.html`.

Michael Moortgat. 1996. Categorial type logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*. Elsevier, Amsterdam.

Glyn V. Morrill. 1994. *Type-logical Grammar: Categorial Logic of Signs*. Kluwer, Dordrecht.

David Pesetsky. 1989. Language-particular processes and the earliness principle. In *GLOW '89*. Manuscript available at `http://www.mit.edu/afs/athena.mit.edu/org/l/linguistics/www/pesetsky.home.html`.

Cecilia Poletto and Jean-Yves Pollock. 1999. On the left periphery of Romance wh-questions. University of Padua, Université de Picardia à Amiens.

Carl Pollard. 1984. *Generalized phrase structure grammars, head grammars and natural language*. Ph.D. thesis, Standford University.

Jean-Yves Pollock, Nicola Munaro, and Cecilia Poletto. 1999. Eppur si muove! In *A Celebration*. MIT Press, Cambridge, Massachusetts. Available at `http://mitpress.mit.edu/chomskydisc/polleto.html`.

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

Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1993. Principles and implementation of deductive parsing. Technical Report CRCT TR-11-94, Computer Science Department, Harvard University, Cambridge, Massachusetts. Available at `http://arXiv.org/`.

Dominique Sportiche. 1995. Sketch of a reductionist approach to syntactic variation and dependencies. In Hector Campos and Paula Kempchinsky, editors, *Evolution and Revolution in Linguistic Theory*. Georgetown University Press, Washington. Reprinted in Dominique Sportiche, *Partitions and Atoms of Clause Structure: Subjects, agreement, case and clitics*. NY: Routledge.

Dominique Sportiche. 1998. *Partitions and Atoms of Clause Structure : Subjects, Agreement, Case and Clitics*. Routledge, NY.

Edward P. Stabler and Edward L. Keenan. 2000. Structural similarity. In A. Nijholt, G. Scollo, T. Rus, and D. Heylen, editors, *Algebraic Methods in Language Processing, AMiLP 2000*, University of Iowa.

Edward P. Stabler. 1997. Derivational minimalism. In Christian Retoré, editor, *Logical Aspects of Computational Linguistics*, pages 68–95. Springer-Verlag (Lecture Notes in Computer Science 1328), NY.

Edward P. Stabler. 1999. Minimalist grammars and recognition. In *Linguistic form and its computation*, Bad Teinach, Germany. Presented at the Final Workshop of SFB340, October 1999. Publication forthcoming.

Willemijn Vermaat. 1999. The minimalist *move* operation in a deductive perspective. In *Proceedings of the Workshop on Resource Logics and Minimalist Grammars, ESSLLI'99*, Utrecht.