A Type-Logical Perspective on Minimalist Derivations

Thomas L. Cornell*

July 17, 1997

Many researchers in the past few years have remarked on the apparent relation between ideas from Chomsky's minimalist program (Chomsky 1995) and fundamental ideas in categorial grammar (e.g., Morrill 1994, Moortgat 1996) We propose to develop ideas from the minimalist program within such a type-logical deductive grammar framework. Simulating minimalist style derivations in a deductive setting turns out to be relatively straightforward, up to a point. Beyond that point, however, problems become quite serious. Nonetheless, the comparison is illuminating, in particular as it sheds some light on the problematical relation of derivational and representational techniques in modern transformational grammar.

The main feature of minimalist derivations which has suggested a connection with categorial grammar is the treatment of feature checking in Chapter 4 of Chomsky (1995). The idea that a feature on a moving phrase might be cancelled against a feature on the structure to which it moves is very reminiscent of the basic cancellation rules of categorial grammar (cf. Stabler forthcoming, 1996). In a type logical setting we would want to implement feature checking as an instance of the application of a rule of use (left rule) for a residual type. To take a somewhat oversimplified example, given a verb which assigns accusative case to its specifier, we can represent the cancellation of the verb's case (K) feature against the K feature of the object as in the following proof.

$$\frac{K \Rightarrow K \quad V \Rightarrow V}{K, K\backslash V \Rightarrow V}\, L\backslash$$

Roughly, the application of the $L\backslash$ rule corresponds to the attachment of the object in the specifier position of the V, and the axiom $K \Rightarrow K$ marks the feature as cancelled.

*This work has benefitted from the help of many people, in particular Dick Oehrle, Ed Stabler and Uwe Mönnich. Much is due to them, but the errors that remain are all my own. Author's address: University of Tübingen, Seminar für Sprachwissenschaft; email: `cornell@sfs.nphil.uni-tuebingen.de`

There are several points which make this approach to minimalist derivations appealing. Most importantly, this offers an explanation grounded in formal, logical considerations for the idea that transformational derivations should be driven by the need to cancel features. In a deductive setting this means neither more nor less than that proofs encode the need to reduce a theorem to axioms.

Another reason to look closely at type logical grammars as a framework for formal investigations of minimalist ideas has to do with the tension in modern day transformational grammar between derivations and representations. The issue revolves around the relative merits of defining syntactic structures by means of movement-based derivations, or chain-based well-formedness conditions. Two things suggest that a type logical setting may clarify these very murky issues. First, this is a framework in which procedural, proof theoretic approaches are balanced by a careful attention to more declarative, model theoretic techniques. On the other hand, the Curry-Howard morphism, and its generalization to other kinds of labeled deduction, offers a somewhat different, but very direct link between derivations and representations. In this paper we will pay some attention in particular to these latter techniques for relating transformational derivations to chains.

As presented in Chomsky (1995), a derivation proceeds from an initially given *numeration*, which is supposed to be a multiset of lexical item occurrences. The numeration is then subjected to the action of a pair of generalized transformations, **Merge** and **Move**, which transform the raw lexical resources of the numeration into an LF representation, delivering an intermediate representation for PF spell-out along the way. In the setting of a grammar logic we want the lexical item instances to be type formulas, and we want to isolate in particular collections of such lexical formulas which can be combined to form sentences. We represent this fundamental decision problem as the question of whether or not a sequent of the form "lexical resources $\Rightarrow$ sentence-category" is derivable. Grafting minimalist ideas into this setting seems fairly natural: we are interested in sequents of the form "Numeration $\Rightarrow C$," where C (complementizer) is the sentence category. Chomsky's definition of the numeration as a multiset even gives us a direct indication of which grammar logic we ought to try to use: **LP**, the Lambek-van Benthem associative and commutative calculus. So our starting point seems clearly given. We begin there and see what kind of trouble we get into.

We present **LP** here in Gentzen style, with nonassociative, noncommutative sequents (so antecedents are binary trees of formulas), with explicit structural rules of Permutation and Association. This will help us track better what these rules actually get used for and where they are needed. We associate type formulas with "proof terms" which record the structure of the proof. These terms are constructed out of variables, which we will instantiate with lexical term assignments. (To save horizontal space we often write the proof term under the type formula.) Axioms are sequents of the form $x : A \Rightarrow x : A$, where x is a fresh variable and A an atomic type.

$$\frac{\Gamma \Rightarrow \genfrac{}{}{0pt}{}{A}{t} \quad \Delta[\genfrac{}{}{0pt}{}{A}{x}] \Rightarrow \genfrac{}{}{0pt}{}{B}{u}}{\Delta[\Gamma] \Rightarrow \genfrac{}{}{0pt}{}{B}{u[t/x]}}\, Cut \tag{1}$$

$$\frac{\Gamma \Rightarrow \genfrac{}{}{0pt}{}{A}{t} \quad \Delta[\genfrac{}{}{0pt}{}{B}{x}] \Rightarrow \genfrac{}{}{0pt}{}{C}{u}}{\Delta[\genfrac{}{}{0pt}{}{B/A}{y}, \Gamma] \Rightarrow \genfrac{}{}{0pt}{}{C}{u[y(t)/x]}}\, L/ \qquad \frac{\Gamma \Rightarrow \genfrac{}{}{0pt}{}{A}{t} \quad \Delta[\genfrac{}{}{0pt}{}{B}{x}] \Rightarrow \genfrac{}{}{0pt}{}{C}{u}}{\Delta[\Gamma, \genfrac{}{}{0pt}{}{A\backslash B}{y}] \Rightarrow \genfrac{}{}{0pt}{}{C}{u[y(t)/x]}}\, L\backslash \tag{2}$$

$$\frac{\Delta[(\genfrac{}{}{0pt}{}{A}{x_1}, \genfrac{}{}{0pt}{}{B}{x_2})] \Rightarrow \genfrac{}{}{0pt}{}{C}{u}}{\Delta[\genfrac{}{}{0pt}{}{A\cdot B}{z}] \Rightarrow \genfrac{}{}{0pt}{}{C}{u[\pi_1 z/x_1, \pi_2 z/x_2]}}\, L\cdot \tag{3}$$

$$\frac{(\Delta, \genfrac{}{}{0pt}{}{A}{x}) \Rightarrow \genfrac{}{}{0pt}{}{B}{u}}{\Delta \Rightarrow \genfrac{}{}{0pt}{}{B/A}{\lambda x.u}}\, R/ \qquad \frac{(\genfrac{}{}{0pt}{}{A}{x}, \Delta) \Rightarrow \genfrac{}{}{0pt}{}{B}{u}}{\Delta \Rightarrow \genfrac{}{}{0pt}{}{A\backslash B}{\lambda x.u}}\, R\backslash \tag{4}$$

$$\frac{\Delta_1 \Rightarrow \genfrac{}{}{0pt}{}{A}{x_1} \quad \Delta_2 \Rightarrow \genfrac{}{}{0pt}{}{B}{x_2}}{(\Delta_1, \Delta_2) \Rightarrow \genfrac{}{}{0pt}{}{A\cdot B}{\langle x_1, x_2\rangle}}\, R\cdot \tag{5}$$

$$\frac{\Delta[((\Gamma_1, \Gamma_2), \Gamma_3)] \Rightarrow B}{\overline{\Delta[(\Gamma_1, (\Gamma_2, \Gamma_3))] \Rightarrow B}}\, A \qquad \frac{\Delta[(\Gamma_2, \Gamma_1)] \Rightarrow B}{\overline{\Delta[(\Gamma_1, \Gamma_2)] \Rightarrow B}}\, P \tag{6}$$

Consider for example the following theorem of **LP**. (In what follows the feature K is used for "case", since we are already using C for "complementizer", and we use W rather than Wh for compactness and uniformity.)

$$\genfrac{}{}{0pt}{}{D\cdot K}{\langle x_1, x_2\rangle}, \genfrac{}{}{0pt}{}{(D\backslash V)/V}{y_1}, \genfrac{}{}{0pt}{}{(K\backslash V)/D}{y_2}, \genfrac{}{}{0pt}{}{D\cdot K}{\langle x_3, x_4\rangle} \Rightarrow \genfrac{}{}{0pt}{}{K\cdot V}{\langle x_2, y_1(y_2(x_3)(x_4))(x_1)\rangle} \tag{7}$$

Intuitively, this represents the construction of a standard transitive verb phrase in the style of Chomsky (1995).[1] A light verb (here assigned the type $(D\backslash V)/V$) selects the external argument (represented here as the first $D \cdot K$, a determiner phrase needing to check a case feature) and the main verb selects a DP complement (the second $D \cdot K$) and assigns accusative case to its specifier. The object DP will have to move from complement to specifier of the lower V, and we will end up with a phrase containing two features still needing to be checked: the V feature of the light verb, and the K feature of the external argument. (We do not illustrate the movement of the main verb to adjoin to the light verb; we will discuss head movement in some detail below.) We assume the sequent comma is right branching, and so we can leave out some unnecessary parentheses.

[1] Or at least, the style arrived at by the end of the book.

$$\dfrac{\dfrac{\dfrac{\overset{D}{x_3} \Rightarrow \overset{D}{x_3} \qquad \dfrac{\dfrac{\overset{K}{x_4} \Rightarrow \overset{K}{x_4} \quad \overset{V}{x_5} \Rightarrow \overset{V}{x_5}}{\overset{K}{x_4}, \overset{K\backslash V}{y_3} \Rightarrow \overset{V}{y_3(x_4)}}\,L\backslash}{\overset{K\backslash V}{y_3}, \overset{K}{x_4} \Rightarrow \overset{V}{y_3(x_4)}}\,\boxed{\text{P,A}}}{(\overset{(K\backslash V)/D}{y_2}, \overset{D}{x_3}), \overset{K}{x_4} \Rightarrow \overset{V}{y_2(x_3)(x_4)}}\,L/}{\overset{(K\backslash V)/D}{y_2}, \overset{D}{x_3}, \overset{K}{x_4} \Rightarrow \overset{V}{y_2(x_3)(x_4)}}\,\boxed{\text{A}}}{\overset{(K\backslash V)/D}{y_2}, \overset{D\cdot K}{\langle x_3,x_4\rangle}, \Rightarrow \overset{V}{y_2(x_3)(x_4)}}\,L\cdot \tag{8}$$

$$\dfrac{(8) \qquad \dfrac{\overset{V}{x_7} \Rightarrow \overset{V}{x_7} \qquad \dfrac{\dfrac{\overset{D}{x_6} \Rightarrow \overset{D}{x_6} \quad \overset{V}{x_8} \Rightarrow \overset{V}{x_8}}{\overset{D}{x_6}, \overset{D\backslash V}{y_4} \Rightarrow \overset{V}{y_4(x_6)}}\,L\backslash}{\overset{D\backslash V}{y_4} \Rightarrow \overset{D\backslash V}{\lambda x_6.y_4(x_6)}}\,R\backslash}{\overset{(D\backslash V)/V}{y_1}, \overset{V}{x_7} \Rightarrow \overset{D\backslash V}{\lambda x_6.y_1(x_7)(x_6)}}\,L/}{\overset{(D\backslash V)/V}{y_1}, \overset{(K\backslash V)/D}{y_2}, \overset{D\cdot K}{\langle x_3,x_4\rangle} \Rightarrow \overset{D\backslash V}{\lambda x_6.(y_1(y_2(x_3)(x_4))(x_6))}}\,Cut \tag{9}$$

$$\dfrac{(9) \qquad \dfrac{\dfrac{\dfrac{\overset{K}{x_2} \Rightarrow \overset{K}{x_2} \qquad \dfrac{\overset{D}{x_1} \Rightarrow \overset{D}{x_1} \quad \overset{V}{x_9} \Rightarrow \overset{V}{x_9}}{\overset{D}{x_1}, \overset{D\backslash V}{y_5} \Rightarrow \overset{V}{y_5(x_1)}}\,L\backslash}{\overset{K}{x_2}, \overset{D}{x_1}, \overset{D\backslash V}{y_5} \Rightarrow \overset{K\cdot V}{\langle x_2,y_5(x_1)\rangle}}\,R\cdot}{(\overset{D}{x_1}, \overset{K}{x_2}), \overset{D\backslash V}{y_5} \Rightarrow \overset{K\cdot V}{\langle x_2,y_5(x_1)\rangle}}\,\boxed{\text{P,A}}}{\overset{D\cdot K}{\langle x_1,x_2\rangle}, \overset{D\backslash V}{y_5} \Rightarrow \overset{K\cdot V}{\langle x_2,y_5(x_1)\rangle}}\,L\cdot}{\overset{D\cdot K}{\langle x_1,x_2\rangle}, \overset{(D\backslash V)/V}{y_1}, \overset{(K\backslash V)/D}{y_2}, \overset{D\cdot K}{\langle x_3,x_4\rangle} \Rightarrow \overset{K\cdot V}{\langle x_2,y_1(y_2(x_3)(x_4))(x_1)\rangle}}\,Cut \tag{10}$$

Figure 1: Sketching a minimalist derivation as a proof in **LP**.

The subproof in (8) represents the result of merging the direct object as complement of the main verb, cancelling the categorial feature against the verb's selection feature. Then the direct object's surviving K feature moves to the specifier position of the verb, which is a checking configuration for formal features. As one would expect, the operation of the generalized transformation **Move** is implemented via a series of applications of the structural rules A and P.

The subproof in (9) merges the lower VP (now just containing the main verb's uncancelled V feature) with the light verb that selects it. Then finally the subproof in (10) merges the subject DP with the light verb as its specifier. We cancel the categorial D feature against that verb's selection feature, leaving the K uncancelled.

Such a derivation would normally proceed by moving the K feature of the subject to attach to Tense or some other case-assigning functional projection. Additionally we might have a W feature on either the subject or the object, which would need to be cancelled against a corresponding feature of the complementizer.

In this example we use our structural rules in three places, only one of which corresponds to an application of **Move** in the corresponding minimalist style derivation. A fourth case is not represented here, mainly because we have presented the numeration in the order in which its contents would be used in a minimalist derivation. In the missing case, an entire feature complex would be moved to the point at which its first feature is used. This should also not be confused with syntactic movement, since it is in principle completely free of constraints, which only begin to apply once we have detached the first feature from the product or, in the case of a functor, applied it to its first argument. The other two cases are forced on us by our use of product types to represent lexical items with multiple features, and by the fact that the logical inference rules can only apply to adjacent, bracketed pairs of formulas. This creates a point of friction with the minimalist program as, for example, the Specifier-Head configuration, which is perhaps the paradigm case of a checking configuration in the minimalist program, does not form an appropriate environment for the application of the $L\backslash$ rule.

One way to reduce this friction might be to adopt a multimodal approach with (at least) two products: one corresponding to "true" structure building, the other a "utility product" used for shuttling features around.

Another direction in which these simple ideas can be refined is suggested by the following. Consider how we might assign values to the proof terms in the goal sequent. At the moment, we would like the proof term associated with the consequent to represent the LF structure we have built. We will assign to the functor categories structure building functions. We use a variant of the notation used in Stabler (1996, forthcoming) for representing minimalist phrase structures. We use the symbols $\lhd, >, \blacktriangleright, \blacktriangleleft$ to indicate which of two subexpressions is the head of the whole (the head is always greater than the nonhead), whether the configuration so formed is a checking configuration (closed triangles represent checking configurations) and whether the configuration is regarded as internal to the X0 or external to it (black triangles are head internal). In traditional terminology, $\lhd$ represents specifier attachment, $>$ complement attachment and $\blacktriangleleft$ and $\blacktriangleright$ represent prefix and suffix attachment, respectively. We prefer this notation to Chomsky's set-theoretic notation at very least for readability reasons, but even more so, in the current context, because it treats all the interior nodes of a phrase structure tree as being labeled with operation symbols that essentially correspond to various sorts of headed concatenation operators. So this notation makes it clear that we are dealing with structures built up from lexical material by the application of product operators, though we will not develop the full fledged multimodal system that this promises in this preliminary investigation.

To arrive at the kinds of trees one finds in Stabler's work, we simply install the first argument in the complement position and the remaining arguments in specifier positions. (We consider X0 internal positions below.) This would give us the following assignments for our two verb formulas, which are essentially structurally identical.

$$(D\backslash V)/V : y_1 = \lambda u_1.\lambda u_2.(u_2 \lhd (v_1 > u_1)) \tag{11}$$

$$(K\backslash V)/D : y_2 = \lambda u_1.\lambda u_2.(u_2 \lhd (v_2 > u_1)) \tag{12}$$

(We will use subscripted v's and d's as placeholders for whatever lexical information may need to be associated with the head position.) In light of these assignments, we can think of $\backslash$ as the residual of the specifier attaching product $\lhd$ and $/$ as the residual of the complement attaching product $>$. It is with an eye towards this move that we have presented a version of **LP** with slashes in both directions: normally, with permutation, these distinctions are irrelevant.

An interesting possibility presents itself when we consider how to assign proof terms to our product categories, the feature complexes $D \cdot K$. We associate them with *chains*, that is,

$$(D \cdot K)_{Subj} : \langle x_1, x_2 \rangle = \langle t_S, d_S \rangle, \tag{13}$$

$$(D \cdot K)_{Obj} : \langle x_3, x_4 \rangle = \langle t_O, d_O \rangle. \tag{14}$$

This move is pretty much forced on us by a type logical perspective. The choice of a product type to represent bundles of formal features is the most natural way to accommodate the resource sensitive treatment of formal features in the minimalist program. But in turn the most natural way to treat a product type as a structure is, well, as a product, that is, a pair or more generally an n-tuple. Intuitively, if we have two types, we ought to have two things belonging to those types. The object which we think of as being both of type K and type D is a complex object, in other words, and the syntax cares about how it was made. In particular, those parts may be associated with different licensing heads, or with the same head but in different structural relations. So together with the procedural idea of feature driven movement we are given as its declarative counterpart the idea of a feature bearing chain.

Particularly interesting in this regard is the fact that the chain associated with a lexical item is, in fact, associated with the lexical item. That is, it is present in the derivation from the very beginning, even before the proof steps corresponding to movement have been executed. This idea, that chains can be formed "presyntactically," is already present in the literature in work by Michael Brody (e.g., Brody 1995), though it does not seem to have received very much attention to date.[2]

So, as the reader can readily verify, the proof term associated with the final VP works out to the following.

$$\langle d_S, (t_S \lhd (v_1 > (d_O \lhd (v_2 > t_O))))\rangle \tag{15}$$

This structure, in which we simply pair an otherwise unattached expression with an otherwise complete tree, has no counterpart in standard syntax. It appears here because we could not use up all our lexical material in a single verb phrase, and because we have to package the consequent of our intuitionistic sequents into a single formula. Whether such a "storage construction" would ever find a use in ordinary syntax remains to be seen.

The main point to observe here is that, given a theory of feature driven movement, the movement potential of a given lexical item is largely given by its lexical makeup. This observation assimilates very nicely with the classical categorial observation that the combinatory potential of a lexical item is essentially given in advance by its syntactic category. And because it is embedded in a grammar logic, this idea receives immediately, via the Curry-Howard morphism, both a declarative and a procedural encoding.

Unfortunately, however appealing these observations may be, serious and immediate problems remain which suggest that there are important ways in which minimalist derivations are *not* like type logical deductions. In order to make something like **Merge** emerge in this deduction, we focused on a very particular sort of proof, which is not in fact very typical of proofs in **LP** and its cousins. Obviously, completely Cut-free proofs are also available. We could, for example, prove the theorem in (7) by first breaking up all lexical feature-complexes with applications of the $L\cdot$ rule, and then merging the individual features in with their licensing functors. However, this allows

[2]Note that Brody assumes that chains are copies of a phrasal structure; here we use traditional traces. In Cornell (1997) we consider chains as sequences of abstract syntactic positions rather than phrasal copies, and this is the syntactic theory that is really lurking in the background here.

several other related theorems as well. One is just the version in which subject and object roles are switched and this is okay, we want that one. But another represents the illegal derivation in which we put t_S in Spec of the higher V, by cancelling the subject's D feature there as before, and we put its "antecedent" d_S in Spec of the lower V, by permuting and reassociating the K feature rightwards.

We could rule this out by placing the traditional sort of c-command requirements on chains as they appear in our proof terms, but up until now we have not used our labeling algebra as a source of constraint; it is used solely to provide a declarative perspective on the derivation. Nor would this be an appropriate use of this constraining device: a lack of independence between derivation and representation spoils their correspondence by the same token. In any case it is a rather important claim of derivational minimalism that such conditions as the c-command condition are unnecessary when we are always adding to the tree at its root. Our Cut-based "derivational deduction" in (8)–(10) imposes just this sort of ordering constraint. As long as we are always combining two lexical formulas at a time, and as long as we use up all the features we can in each subproof then the ill formed chains observed above will simply never occur. However, we cannot simply stipulate that we are interested only in proofs of a certain kind, and still claim the benefits of working in a grammar logic.

We can construct at least a partial solution by making use of Moortgat's unary modality operators $\diamond$ and $\Box^{\downarrow}$. (Cf. Moortgat (1996). The basic idea we use here is due originally to Morrill (1994).) The logical rules for this "unary product" and its residual are given below.

$$\frac{\Delta[(A)^{\diamond}] \Rightarrow B}{\Delta[\diamond A] \Rightarrow B}\, L\diamond \qquad \frac{\Delta[A] \Rightarrow B}{\Delta[(\Box^{\downarrow} A)^{\diamond}] \Rightarrow B}\, L\Box^{\downarrow} \tag{16}$$

$$\frac{\Delta \Rightarrow A}{(\Delta)^{\diamond} \Rightarrow \diamond A}\, R\diamond \qquad \frac{(\Delta)^{\diamond} \Rightarrow A}{\Delta \Rightarrow \Box^{\downarrow} A}\, R\Box^{\downarrow} \tag{17}$$

We also make use of the following two "weak distributivity" postulates governing the interaction of the unary and binary modality.

$$\frac{\Delta[((\Gamma_1)^{\diamond}, \Gamma_2)] \Rightarrow A}{\Delta[((\Gamma_1, \Gamma_2))^{\diamond}] \Rightarrow A}\, K1 \qquad \frac{\Delta[(\Gamma_1, (\Gamma_2)^{\diamond})] \Rightarrow A}{\Delta[((\Gamma_1, \Gamma_2))^{\diamond}] \Rightarrow A}\, K2 \tag{18}$$

Suppose now that we revise our lexical type formulas so that the heads of functors are marked with a $\Box^{\downarrow}$. We furthermore insist that every functor be contained in a $(\cdot)^{\diamond}$ bracketing which contains no other functor not contained in a smaller such bracketing. More simply: a $\diamond$ must intervene between every pair of functor categories. Only certain such bracketings can lead to successful derivations, namely those which capture enough lexical material in their scope to cancel all antecedent features of the functor. Once we

$$
\dfrac{D \Rightarrow D \qquad \dfrac{\dfrac{\dfrac{K \Rightarrow K \qquad \dfrac{\dfrac{\dfrac{\text{etc.}}{V, W \Rightarrow V \cdot W}}{(\Box^{\downarrow}V)^{\Diamond}, W \Rightarrow V \cdot W}\, L\Box^{\downarrow}}{(\Box^{\downarrow}V, W)^{\Diamond} \Rightarrow V \cdot W}\, K1}{((K, K\backslash\Box^{\downarrow}V), W)^{\Diamond} \Rightarrow V \cdot W}\, L\backslash}{(K\backslash\Box^{\downarrow}V, K, W)^{\Diamond} \Rightarrow V \cdot W}\, P, A}{(K\backslash\Box^{\downarrow}V, K \cdot W)^{\Diamond} \Rightarrow V \cdot W}\, L\cdot}{\dfrac{(((K\backslash\Box^{\downarrow}V)/D, D), K \cdot W)^{\Diamond} \Rightarrow V \cdot W}{\dfrac{((K\backslash\Box^{\downarrow}V)/D, D, K \cdot W)^{\Diamond} \Rightarrow V \cdot W}{((K\backslash\Box^{\downarrow}V)/D, D \cdot K \cdot W)^{\Diamond} \Rightarrow V \cdot W}\, L\cdot}\, A}\, L/
$$

Figure 2: A derivation within a domain of freedom.

have managed to cancel all the antecedent features of the functor, possibly after some application of $K1$ or $K2$, we will be able to apply the $L\Box^{\downarrow}$ rule to remove the modals, now that their work is done. The point of this operation is that the $\Diamond$ modal effectively seals off its scope from the application of A or P. Now the entire subconstruction can be permuted or reassociated, and its contents can also move freely within the scope of the operator, but structural rules, limited to binary contexts, cannot apply across this unary barrier. Then the embedded $\Box^{\downarrow}$ operator functions as a key to unlock this barrier, but a key which can only be accessed once all the other features of the head have been used up. In figure 2, we assume that the object is a Wh phrase, to illustrate the use of the $K1$ rule.

This is a step in the right direction. However, it involves considerable use of force, since now we must assume a degree of structure in the numeration which is apparently not required in ordinary minimalist derivations. In fact, the $\Diamond$ domains have essentially the effect of selecting a particular order of processing for elements in a numeration, so the freedom of choice that is supposed to hold in the introduction of numeration items into the derivation is lost. To some extent all of this control information is lexically based, by virtue of the fact that every $\Diamond$ must balance against a $\Box^{\downarrow}$. Still it does not appear to us as neat a solution to the problem of control as the minimalist requirement that all transformations should extend their target, and work remains to be done.

Let us turn now to problems posed by head movement constructions. We take as our example here an artificial language, the "copy language" over the alphabet $\{0, 1\}$. We will derive the strings of this (non-context free) language using head movement in a minimalist derivational style. The idea is to simulate in this artificial language the formation of head (e.g., verb) clusters in natural languages. These are supposed to pose serious problems for representational variants of GB and minimalism, and so they are of special interest in the study of the relations between derivational and representational approaches.

We will assume, as a point of departure, a theory in which head movement is triggered by the selectional requirements of the host head. As overt XP movement is triggered by strong formal features, we will follow Stabler (following Koopman 1994) in treating overt head movement as triggered by strong selection features. A theory of selection is conspicously missing from Chomsky (1995). **Merge** is supposed to be free to apply at any time, and is not required to cancel any features. To achieve a more uniformly resource-conscious theory we have been tacitly assuming all along a categorial style theory of selection to go with our minimalist style theory of attraction. Up to this point those systems have been essentially modular: selection is concerned with categorial features and attraction with "inflectional" features like case. Here, however, those systems overlap, and not without some tricky formal problems.

For our lexicon for the copy language we make use of four atomic features, $\{S, F, A, B\}$. From these we construct the following lexical items, with term assignments as given.

$$S/F : \lambda u_1.((\epsilon \blacktriangleright \pi_1 u_1) > \pi_2 u_1) \tag{19}$$

$$((A\backslash F)/F)_i : \lambda u_1.\lambda u_2.\langle (u_2 \lhd (t_i > \pi_1 u_1)), (0 \blacktriangleright \pi_2 u_1)\rangle \tag{20}$$

$$((B\backslash F)/F)_i : \lambda u_1.\lambda u_2.\langle (u_2 \lhd (t_i > \pi_1 u_1)), (1 \blacktriangleright \pi_2 u_1)\rangle \tag{21}$$

$$F_i : \langle t_i, \epsilon\rangle \tag{22}$$

$$A : 0 \tag{23}$$

$$B : 1 \tag{24}$$

The strings of the copy language will be the yield of the proof terms for category S. Intuitively, the various F categories form a recursive spine, with A and B specifiers which "agree" with their heads on the symbol they dominate. We consider all antecedent occurrences of F to be strong selectors. This has the effect of cyclically extracting every second occurrence of a 1 or 0 and collecting them under the head of the matrix S.

As usual the terms associated with nonfunctors are chains. Up until now, however, the length of the chain has been a function of the number of features in the type or, to put it the other way around, the type formula has accurately reflected the structure of the chain: a chain is a tuple, so its type is a tuple. Here we have a proof object with the formal properties of a nontrivial chain but which is assigned a single type. This works because of the tight connection between complement taking and complex head formation. For example, Stabler (1996) treats them as part of the same algebraic operation.

So now, for example, the following is an **NL** theorem, that is, we could derive it even in the absence of any structural rules.

$$\underset{y_1}{S/F}, \underset{x_1}{A}, \underset{y_2}{(A\backslash F)/F}, \underset{x_2}{B}, \underset{y_3}{(B\backslash F)/F}, \underset{x_3}{F} \Rightarrow \underset{y_1(y_2(y_3(x_3)(x_2))(x_1))}{S} \tag{25}$$

With the term assignment in (19)–(24) this evaluates to the following, which yields the string 0101.

$$((\epsilon \blacktriangleright (0 \blacktriangleright (1 \blacktriangleright \epsilon))) > (0 \lhd (t_0 > (1 \lhd (t_1 > t_\epsilon))))) \tag{26}$$

It might be preferable from a formal standpoint to make our treatment of head chains more consistent with our treatment of XP chains. That is, if there are two elements of the head chain, each should have a type, a grammatical resource which is used in the construction of that chain. The problem is that there is no compelling theory of head movement which identifies it with some feature distinct from the category of the head. For the moment we will propose a formal solution in advance of any theory, using X as our mystery feature (which might well simply be a second copy of F). In this setting we no longer require projection functions in our proof terms: in a derivation, the head chain is broken up by $L\cdot$ as with XP chains. However, now the sequent corresponding to (25) no longer has a proof in **NL**: we require associativity to reassociate a feature from the rest of its lexical bundle to the functor against which it cancels.

$$(S/X)/F : \lambda u_1.\lambda u_2.((\epsilon \blacktriangleright u_2) > u_1) \tag{27}$$

$$(((A\backslash(F\cdot X))/X)/F)_i : \lambda u_1.\lambda u_2.\lambda u_3.\langle (u_3 \lhd (t_i > u_1)), (0 \blacktriangleright u_2)\rangle \tag{28}$$

$$(((B\backslash(F\cdot X))/X)/F)_i : \lambda u_1.\lambda u_2.\lambda u_3.\langle (u_3 \lhd (t_i > u_1)), (1 \blacktriangleright u_2)\rangle \tag{29}$$

$$(F\cdot X)_i : \langle t_i, \epsilon\rangle \tag{30}$$

$$A : 0 \tag{31}$$

$$B : 1 \tag{32}$$

We must distinguish features which attract heads from features which attract phrases. As with $\lhd$ and $>$, then, we might treat the / over the X feature as the residual of the suffix attaching $\blacktriangleright$ product. Then a tuple of features corresponds to *two* chains, whose members are interleaved; but the two distinct chains are easily recovered by considering the type of each link. One further point about the representation of chains needs to be made: the initial position of the X0 chain is always embedded in the XP. So in fact each of the 2-tuples depicted above represent at once a one-position XP chain and a two-position X0 chain.

We have at various points alluded to a multimodal version of this system, which is lurking behind the structure forming operators of the term algebra. Such a system would give us much more detailed control over derivations, control which we will probably need to deal with locality problems adequately. But this system is clearly going to be quite complex. One needs four products for the structure forming operations, a lexical product, for forming chains, and a "utility" product for converting checking configurations like Spec-Head to pairs which can trigger a $L/$ rule. These various products interact in very complex ways, and much of the complexity has to do with low-level details, like overcoming the structural distance between a specifier and its head, which are not of much linguistic interest. Therefore we thought it best to start with the simplest ideas, since these already present us with a good collection of hard problems to solve. Nonetheless it is already fairly clear in what directions these simple ideas can be elaborated. We can continue to follow the basic technique of using the desired structures as proof terms, gradually unpacking them so that less of the syntax is done there and more is done in the type system. In effect, we follow a model theoretic program in which the proof terms represent the structures we desire to isolate and reason about, and we develop linguistic and proof theoretic resources to accomplish this. However, the flow of ideas is not at all one-way. For example, it is the resource-consciousness of the proof system that suggests the necessity of Brody-style presyntactic chain formation.

References

Brody, M. (1995). *Lexico-Logical Form*, MIT Press, Cambridge, MA.

Chomsky, N. (1995). *The Minimalist Program*, MIT Press.

Cornell, T. (1997). Representational minimalism, *Technical Report 83*, SFB 340, University of Tübingen.

Koopman, H. (1994). Licensing heads, *in* D. Lightfoot and N. Hornstein (eds), *Verb Movement*, Cambridge University Press, Cambridge, pp. 261–296.

Moortgat, M. (1996). Categorial type logics, *in* J. van Benthem and A. ter Meulen (eds), *Handbook of Logic and Language*, Elsevier, chapter 2.

Morrill, G. (1994). *Type Logical Grammar*, Kluwer, Dordrecht.

Stabler, E. P. (1996). Derivational minimalism, *Proceedings of LACL-96*, Nancy, France.

Stabler, E. P. (forthcoming). *Acquiring and Parsing Languages with Movement*, Basil Blackwell, Oxford.