A Minimal Logic for Minimalism

Alain Lecomte and Christian Retoré
Alain.Lecomte@upmf-grenoble.fr
retore@irisa.fr

September 12, 1999
The third Tbilisi Symposium on Language, Logic and Computation
Batumi, Georgia

1 Chomsky's Minimalist Program

1.1 The Framework

According to Chomsky [1], language must be studied by taking his 'legibility conditions' as a starting point, that means that we have to explain how it is constrained by our mental architecture and our sensorimotor apparatus. In an 'ideal' situation, words (or more precisely, lexical items) would have no other features than those interpreted at the interfaces: properties of sound and meaning, and the 'computational system' attached to the language would use these features and only these ones, no other element would be introduced. But, as Chomsky says, we know that this is not the current situation and that there are some 'imperfections' with regards to this ideal picture: *commonly, phrases are interpreted in positions other than those where they are heard, though in analogous expressions these positions are occupied, and interpreted under natural conditions of locality* (**dislocation property**), and there are *uninterpretable features* like *Structural Case*. Of course, the situation would be nicer if we could reduce these two imperfections into only one: dislocation would be caused by the need for deleting uninterpretable features. Chomsky expresses this idea in terms of *Suicidal Greed*: features are attracted by *attractors* and attractors are deleted when they have attracted their matching feature.

1.2 Minimalist Grammars

E. Stabler ([9], [10]) gives a precise formulation of the Minimalist Principles in his *Minimalist Grammars*. In these grammars, he defines *Merge* and *Move* as operations between trees (*Merge*) or on trees (*Move*) the leaves of which are labelled by lists of features. The goal of a derivation in this system is to delete all the uninterpretable features. The whole derivation yields a tree where some leaves are simply labelled by phonetic features and others by logical ones.
In these grammars, each item of the lexicon consists in a sequence of features, which are divided as follows:

- Phonetic features for example */speaks/,/linguist/,/some/,...*
- Semantic features for example *(speaks),(linguist),(some),...*
- Syntactic or formal features:
  - categorial features (categories) involved in *merge*:
    $\text{BASE} = \{\mathtt{c}, \mathtt{t}, \mathtt{v}, \mathtt{d}, \mathtt{n}, \ldots\}$
  - functional features involved in *move*:
    $\text{FUN} = \{\overline{\mathtt{k}}, \overline{\mathtt{K}}, \overline{\mathtt{wh}}, \ldots\}$

These sequences are described by the following regular expression:
LABEL = SELECT* (LICENSORS) SELECT* BASE LICENSEES P^* I^*

- P phonetic features
- I semantic features
- SELECT = $\{=\mathtt{b}, =\mathtt{B}, \mathtt{B}=|\mathtt{b} \in$ BASE$\}$ select a category
- LICENSEES = $\{-\mathtt{x}|\mathtt{x} \in$ FUN$\}$ needs a move feature
- LICENSORS = $\{+\mathtt{x}, +\mathtt{X}|\mathtt{x} \in$ FUN$\}$ provides a move feature

Merge is defined between two T-markers u and t the head of u starting with $=\mathtt{x}$ and the head t starting with x with $x \in$ BASE. Let u' (resp. t') denote u (resp. t) in which the $=\mathtt{x}$ (resp. x) feature starting the head is cancelled,

- if u is a lexical item then the resulting tree is $u' < t'$ (so u' is the head and is on the left)
- otherwise the resulting tree is $t' > u'$ (so u' is the head in this case as well, but it is on the right)

Roughly speaking, movement is defined as follows: assume that at the leftmost position (spec* position) we have a $+\mathtt{x}$ and that at the rightmost (comp$^+$ position) we have a $-\mathtt{x}$: then the movement takes the whole constituent having $-\mathtt{x}$ as a head and moves it to the leftmost position (spec* position).

1.3 MP and Resource Logics

We think that Chomsky's position may be better expressed in terms of a resource consumption logic. In fact, *Suicidal Greed* may be formulated in a resource logic as: some feature *consumes* another one and after deletes.
In this paper, we try to show how the two operations which are now fundamental in Chomsky's view: *Merge* and *Move* are very conveniently recast in logical terms. This has as a consequence that these two operations no longer appear as the primitive operations: more primitive ones are arising. This is one of the advantages of the logical analysis : to make objects decomposed in more primitive parts and recomposed by very simple operations. Other researchers have presented works with a similar purpose, we may particularly notice W. Vermaat ([11]) T. Cornell ([2]), and one of the authors ([4]). They are using the framework known as Multimodal Categorial Grammar, in the spirit of [6]. In their approach, features are conceived like modalities and components are assembled by means of various composition modes. We think that the introduction of modalities and of the complex machinery associated does not pursue the goal of simplicity that we would like to reach in order to express so primitive mechanisms. It is the reason why we do not use modalities in our framework. That does not entail we should not use different composition modes provided that we introduce the fewest as possible. We shall see that in fact two composition modes seem to be necessary but one of the two remains implicit.
Vermaat's system has been shown equivalent to the Stabler's grammars. It is also the case for our system.

2 A logical analysis of Merge

The operation *Merge* is legitimated by the following observation: an object which already exists (either from the lexicon or by previous construction) has some property which can only be satisfied by another object, that will put these two objects together, and then, when the property is satisfied, we shall consider it as "inactive".
If we provisionally introduce here two connectives: $\bullet$ and $/$, with rules given like in the Lambek calculus, we can describe this fact in the following way.

[2] *Merge (2)* : $\phi/F, F \bullet \psi \vdash \phi \bullet \psi$

where $\bullet$ is *associative* and *non-commutative.* The proof is, in the sequent calculus:

$$\dfrac{\dfrac{F \vdash F \qquad \dfrac{\phi \vdash \phi \quad \psi \vdash \psi}{\phi, \psi \vdash \phi \bullet \psi}[R\bullet]}{\phi/F, F, \psi \vdash \phi \bullet \psi}[L/]}{\phi/F, F \bullet \psi \vdash \phi \bullet \psi}[L\bullet]$$

or, in the Natural Deduction style:

$$\dfrac{F \bullet \psi \qquad \dfrac{\dfrac{\phi/F \quad [F]^1}{\phi}[/E] \qquad [\psi]^1}{\phi \bullet \psi}[\bullet I]}{\phi \bullet \psi}[\bullet E]^1$$

We can provisionally conclude that / and $\bullet$ are primitive (more primitive than *Merge*).
We know that, for logical reasons, we also have the connective \, in such a way that:

$$A \bullet B \vdash C \Leftrightarrow A \vdash C/B$$
$$A \bullet B \vdash C \Leftrightarrow B \vdash A\backslash C$$

Of course, the \ operator is not superfluous. With the /, we were able to attach a syntactic object to the right of another. If we follow the LCA convention (Kayne 94), that corresponds to the attachment of a complement. Because it is assumed in this framework that trees are binary, if the head selects a new syntactic object, it will be attached on the left, and therefore we shall have necessarily to use this \. That simply duplicates the two rules for /, leading to analogous rules for \.

3 Phonological interpretations

According to Chomsky,

> We understand L to be a device that generates expressions EXP, EXP=<PHON, SEM>, where PHON provides the "instructions" for sensorimotor systems and SEM for systems of thought.

If the ideal picture was the case we should have only expressions consisting in <PHON, SEM> pairs, **but** uninterpretable features bring a **third** component so that we have: EXP=<PHON,UNINT,SEM>. In the following rules, we only include phonetic features: they are considered labels.

$$\dfrac{\Delta \vdash \beta : A \quad \Gamma, \gamma : B, \Gamma' \vdash \delta : C}{\Gamma, \alpha : B/A, \Delta, \Gamma' \vdash \delta[\alpha\beta/\gamma]C}[L/] \qquad \dfrac{\Delta \vdash \beta : A \quad \Gamma, \gamma : B, \Gamma' \vdash \delta : C}{\Gamma, \Delta, \alpha : A\backslash B, \Gamma' \vdash \delta[\beta\alpha/\gamma]C}[L\backslash]$$

$$\dfrac{\Gamma, \alpha : A, \beta : B, \Gamma' \vdash C}{\Gamma, \alpha\beta : A \bullet B, \Gamma' \vdash C}[L\bullet] \qquad \dfrac{\Gamma \vdash \alpha : A \quad \Delta \vdash \beta : B}{\Gamma, \Delta \vdash \alpha\beta : A \bullet B}[R\bullet]$$

$$\alpha : A \vdash \alpha : A \; [axiom] \qquad \dfrac{\Gamma \vdash \alpha : A \quad \Delta, x : A, \Delta' \vdash \gamma : C}{\Delta, \Gamma, \Delta' \vdash \gamma[\alpha/x] : C}[cut]$$

These rules can be restated as Natural Deduction rules:

$$\dfrac{\Gamma \vdash x : A/B \quad \Delta \vdash y : B}{\Gamma; \Delta \vdash xy : A}[/E] \qquad \dfrac{\Delta \vdash y : B \quad \Gamma \vdash x : B\backslash A}{\Delta; \Gamma \vdash yx : A}[\backslash E]$$

$$\frac{\Gamma \vdash \alpha : A \bullet B \quad \Delta; x : A; y : B; \Delta' \vdash \gamma : C}{\Delta; \Gamma; \Delta' \vdash \gamma[\alpha/xy]C}[\bullet E] \qquad \frac{\Gamma \vdash x : A \quad \Delta \vdash x : B}{\Gamma, \Delta \vdash xy : A \bullet B}[\bullet I]$$

So, let us start with the following lexicon:

$$\begin{array}{lll} \textit{reads} & ::= & \vdash \textit{reads} : ((\overline{\mathtt{k}}\backslash \mathtt{vp})/\mathtt{d}) \\ a & ::= & \vdash a : ((\mathtt{d} \bullet \overline{\mathtt{k}})/\mathtt{n}) \\ \textit{book} & ::= & \vdash \textit{book} : \mathtt{n} \end{array}$$

with $\overline{\mathtt{k}}$ a formal feature (*case*) and $\mathtt{d}$, $\mathtt{n}$, $\mathtt{vp}$ categorial features (all these features being uninterpretable). We can have the following derivation in order to build a new syntactic object:

$$\dfrac{\dfrac{\vdash a : ((\mathtt{d} \bullet \overline{\mathtt{k}})/\mathtt{n}) \quad \vdash \textit{book} : \mathtt{n}}{\vdash a\ \textit{book} : \mathtt{d} \bullet \overline{\mathtt{k}}}[/E] \quad \dfrac{\dfrac{\vdash \textit{reads} : ((\overline{\mathtt{k}}\backslash \mathtt{vp})/\mathtt{d}) \quad x : \mathtt{d} \vdash x : \mathtt{d}}{x : \mathtt{d} \vdash \textit{reads}\ x : (\overline{\mathtt{k}}\backslash \mathtt{vp})}[/E] \quad y : \overline{\mathtt{k}} \vdash y : \overline{\mathtt{k}}}{x : \mathtt{d}, y : \overline{\mathtt{k}} \vdash \textit{reads}\ xy : (\overline{\mathtt{k}}\backslash \mathtt{vp}) \bullet \overline{\mathtt{k}}}[\bullet I]}{\vdash \textit{reads}\ a\ \textit{book} : (\overline{\mathtt{k}}\backslash \mathtt{vp}) \bullet \overline{\mathtt{k}}}[\bullet E]$$

It is important to notice two things:

- in the lexicon, words are not associated with formulae (like it is the case in usual categorial grammars, and notably in standard Lambek grammars), but with sequents: they are therefore considered *extra-logical axioms*, which are labelled by the phonetic feature of the word itself,
- because $\bullet$ is non commutative, there's no kind of move here. For instance, in the rule [$\bullet$E], α substitutes for the concatenation of x and y, which label the types A and B, which are adjacent in the LHS of the sequent and which can be neither permuted nor displaced.

4 A logical analysis of Attract+Move

Still according to Chomsky: attraction (hence movement) is driven by the need to delete an uninterpretable feature F (we call it the attractor). Moreover, 'the attractor F in the label L of the target β locates the closest F' in its domain, attracting it to the multi-lexical item of F'. We can assume in our framework that attractors occur as negative polarity items ($F\backslash$), attracted features as positive ones, and that the so called Domain of F is the result of several merge steps, thus resulting in a sign like:

$$F\backslash\phi_1 \bullet \phi_2 \bullet ... \bullet F \bullet ... \bullet \phi_n$$

but dislocation needs to *relax* the order ($\bullet$) of the hypotheses. For instance,

$$\textit{reads}\ a\ \textit{book} : (\overline{\mathtt{k}}\backslash \mathtt{vp}) \bullet \overline{\mathtt{k}}$$

cannot be reduced.
This enforces us to introduce a second product, $\otimes$ between types, which is commutative, the structural counterpart of which being ',' (whereas the structural counterpart of $\bullet$ is ';'). By doing so, we are working inside the calculus pCLL ('partially commutative linear logic') designed by P. de Groote ([3]). The two products are able to communicate through the entropy rule:

$$\frac{\Gamma[(\Delta_1; \Delta_2)] \vdash A}{\Gamma[(\Delta_1, \Delta_2)] \vdash A}$$

and we have the following $\otimes$-*elimination rule*:

$$\frac{\Gamma \vdash \alpha : A \otimes B \quad \Delta, x : A, y : B \vdash \gamma : C}{\Gamma, \Delta \vdash \gamma[\alpha/\{x, y\}] : C}[\otimes E]$$

where $\gamma[\alpha/\{x,y\}]$ means the *substitution of* α *to the unordered set* $\{x,\ y\}$ that is the simultaneous substitution of α for both x and y, *no matter the order between* x *and* y *is.*
In this new framework, we can assume the following lexicon:

$$
\begin{array}{lll}
\textit{reads} & ::= & \vdash \textit{reads} : ((\overline{\mathtt{k}}\backslash\mathtt{vp})/\mathtt{d}) \\
a & ::= & \vdash a : ((\mathtt{d}\otimes\overline{\mathtt{k}})/\mathtt{n}) \\
\textit{book} & ::= & \vdash \textit{book} : \mathtt{n}
\end{array}
$$

and we may now build up a new syntactic object by means of the following derivation: [5]

$$
\dfrac{\dfrac{\vdash a : ((\mathtt{d}\otimes\overline{\mathtt{k}})/\mathtt{n}) \quad \vdash \textit{book} : \mathtt{n}}{\vdash a\ \textit{book} : \mathtt{d}\otimes\overline{\mathtt{k}}}[/E] \quad \dfrac{\dfrac{y:\overline{\mathtt{k}} \vdash y:\overline{\mathtt{k}} \quad \dfrac{\vdash \textit{reads} : ((\overline{\mathtt{k}}\backslash\mathtt{vp})/\mathtt{d}) \quad x:\mathtt{d}\vdash x:\mathtt{d}}{x:\mathtt{d}\vdash \textit{reads}\ x : (\overline{\mathtt{k}}\backslash\mathtt{vp})}[/E]}{y:\overline{\mathtt{k}}; x:\mathtt{d}\vdash y\ \textit{reads}\ x : \mathtt{vp}}[\backslash E]}{y:\overline{\mathtt{k}}, x:\mathtt{d}\vdash y\ \textit{reads}\ x : \mathtt{vp}}[\textit{entropy}]}{\vdash a\ \textit{book reads}\ a\ \textit{book} : \mathtt{vp}}[\otimes E]
$$

We get an order in the label *even if we have got through entropy.* It is so because when using [/ E] and [\ E], we necessarily order the labels, and this order is then recorded inside the label and never destroyed, even when using the entropy rule: at this moment, it is only the order *on hypotheses* which is relaxed. We obtain a subsystem of pCLL by simply restricting the proof space to proofs which only contain some particular kinds of step. Let us call $\mathcal{MG}$-proofs those proofs. We have:

Definition 1 *$\mathcal{MG}$-proofs contain only three kinds of steps:*

- *implication steps (elimination rules for / and \)*
- *tensor steps (elimination rule for $\otimes$)*
- *entropy steps (entropy rule)*

We can moreover assume:

Definition 2 *A lexical entry consists in an axiom $\vdash w : \mathcal{T}$ where $\mathcal{T}$ is a type:*

$$((F_2\backslash(F_3\backslash...(F_n\backslash(G_1\otimes G_2\otimes...\otimes G_m\otimes A))))/F_1)$$

where:

- *m and n can be any number greater than or equal to 0,*
- *F_1, ..., F_n are attractors,*
- *G_1, ..., G_m are features,*
- *A is the resulting category type*

But that's not all because we need to express the fact that 'the attractor F in the label L of the target β locates the *closest* F' in its domain. This simply corresponds to the following restriction.

Definition 3 (Shortest Move) *: A $\mathcal{MG}$-proof is said to respect the shortest move condition if it is such that hypotheses are discharged in a First In, First Out order.*

5 Examples

5.1 SVO languages

Let us look at a very simple example, corresponding to an elementary sentence in a SVO language:

```
every linguist speaks some language
```

Before giving the proof, we give the lexicon which is used, together with the translation from Stabler's labels into $\mathcal{MG}$-axioms.

entry	*Stabler′stype*	*label* : *type*
every	$=\mathtt{n}\ \mathtt{d}\ -\overline{\mathtt{k}}$ *every*	*every* : $((\overline{\mathtt{k}}\otimes\mathtt{d})/\mathtt{n})$
some	$=\mathtt{n}\ \mathtt{d}\ -\overline{\mathtt{k}}$ *some*	*some* : $((\overline{\mathtt{k}}\otimes\mathtt{d})/\mathtt{n})$
language	$\mathtt{n}$ *language*	*language* : $\mathtt{n}$
linguist	$\mathtt{n}$ *linguist*	*linguist* : $\mathtt{n}$
speaks	$=\mathtt{d}\ +\overline{\mathtt{k}}\ =\mathtt{d}\ \mathtt{v}$ *speaks*	*speaks* : $((\overline{\mathtt{k}}\backslash(\mathtt{d}\backslash\mathtt{v}))/\mathtt{d})$
(*tense*)	$=\mathtt{v}\ +\overline{\mathtt{k}}\ \mathtt{t}$	$((\overline{\mathtt{k}}\backslash\mathtt{t})/\mathtt{v})$
(*comp*)	$=\mathtt{t}\ \mathtt{c}$	$(\mathtt{t}\backslash\mathtt{c})$

We begin by showing the proof in ND format, and by reversing the proof, we show that we get a tree structure similar to T-markers. For reasons of size of the proof, it will be cut off into two pieces, the first piece gives a reduction of

```
speaks some language
```

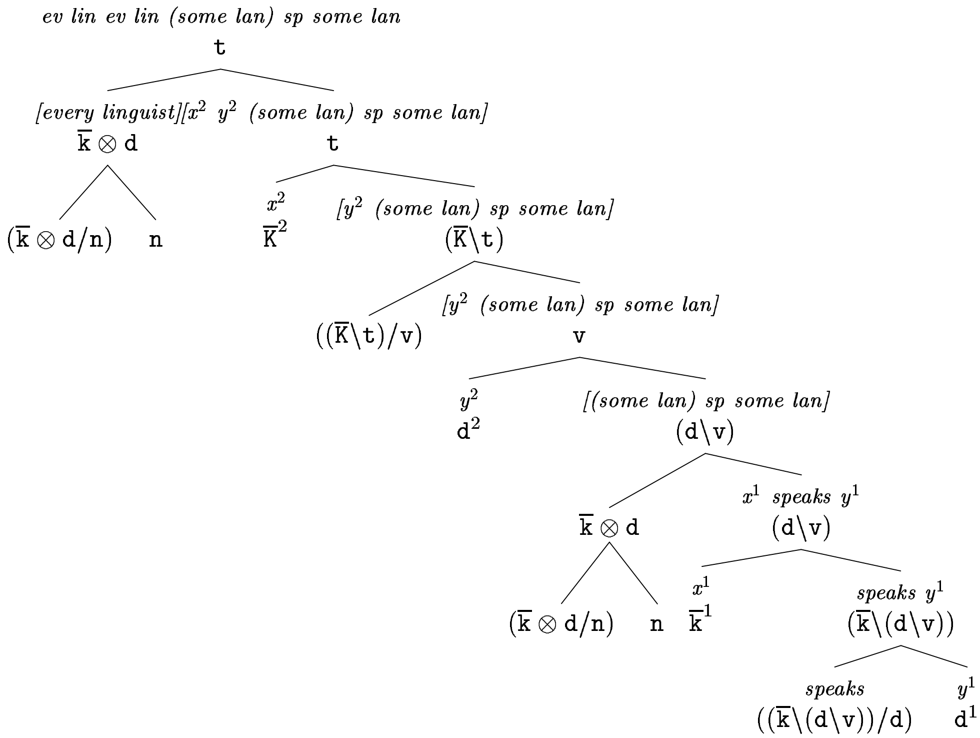
to $\mathtt{v}$, and the second piece shows the continuation of the proof, by using the conclusion thus obtained.

$$\dfrac{\begin{matrix}y^2\\ \mathtt{d}^2\end{matrix} \quad \dfrac{\dfrac{\begin{matrix}\text{some}\\ ((\overline{\mathtt{k}}\otimes\mathtt{d})/\mathtt{n})\end{matrix} \quad \begin{matrix}\text{language}\\ \mathtt{n}\end{matrix}}{\begin{matrix}\textit{some language}\\ \overline{\mathtt{k}}\otimes\mathtt{d}\end{matrix}}[/E] \quad \dfrac{\begin{matrix}x^1\\ \overline{\mathtt{k}}^1\end{matrix} \quad \dfrac{\begin{matrix}\text{speaks}\\ ((\overline{\mathtt{k}}\backslash(\mathtt{d}\backslash\mathtt{v}))/\mathtt{d})\end{matrix} \quad \begin{matrix}y^1\\ \mathtt{d}^1\end{matrix}}{\begin{matrix}\textit{speaks } y^1\\ (\overline{\mathtt{k}}\backslash(\mathtt{d}\backslash\mathtt{v}))\end{matrix}}[/E]}{\begin{matrix}x^1\ \textit{speaks}\ y^1\\ (\mathtt{d}\backslash\mathtt{v})\end{matrix}}}{\begin{matrix}\textit{(some language)}\\ \textit{speaks some language}\\ (\mathtt{d}\backslash\mathtt{v})\end{matrix}}[\otimes E]^1}{\begin{matrix}y^2\ \textit{(some language)}\\ \textit{speaks some language}\\ \mathtt{v}\end{matrix}}[\backslash E]$$

And the continuation of the proof is :

$$\dfrac{\dfrac{\begin{matrix}\text{every}\\ ((\overline{\mathtt{k}}\otimes\mathtt{d})/\mathtt{n})\end{matrix} \quad \begin{matrix}\text{linguist}\\ \mathtt{n}\end{matrix}}{\begin{matrix}\textit{every linguist}\\ \overline{\mathtt{k}}\otimes\mathtt{d}\end{matrix}}[/E] \quad \dfrac{\begin{matrix}x^2\\ \overline{\mathtt{K}}^2\end{matrix} \quad \dfrac{\begin{matrix}\emptyset\text{: tense}\\ ((\overline{\mathtt{K}}\backslash\mathtt{t})/\mathtt{v})\end{matrix} \quad \begin{matrix}y^2\ \textit{(some language)}\\ \textit{speaks some language}\\ \mathtt{v}\end{matrix}}{\begin{matrix}y^2\ \textit{(some language)}\\ \textit{speaks some language}\\ (\overline{\mathtt{K}}\backslash\mathtt{t})\end{matrix}}[/E]}{\begin{matrix}x^2\ y^2\ \textit{(some language) speaks some language}\\ \mathtt{t}\end{matrix}}[\backslash E]}{\begin{matrix}\textit{every linguist every linguist}\\ \textit{(some language) speaks some language}\\ \mathtt{t}\end{matrix}}[\otimes E]^2$$

Let us see the tree we obtain.



Comments: a transitive verb like *speaks* has a categorial feature looking for a d on the right, and a functional feature $\overline{k}$ (*case*) on its left. These two demands are satisfied by two hypotheses. By elimination of / and then of \ (here analogous to *merge*), the labels of these hypotheses are incorporated into the verb. But it is only in a second step that these hypotheses are discharged by means of elimination of $\otimes$. Because objective case is weak, only the semantic part of the object is substituted to x^1, and the content of the object is substituted to y^1, thus resulting in the node (*some language*) *speaks some language* : (d\v). The integration of *tense* (or *inflection*) makes $\overline{K}$ to occur, and it will be cancelled (or *checked*) only by a new d requiring a case. But this time, because nominative case is strong, the whole content of the subject is attracted to the highest position, that means that the two variables x^2 and y^2 are substituted by the same content.
We may now call "move", the kind of line we can draw in such a tree from a variable y^i to a variable x^i which has the same index. The content that "moves" is determined by the content of the node which makes possible to substitute the two variables.

The conventions for reading the result are:

- all strings not inside parentheses or slashes must be read at the same time as phonetic and semantic features,
- any second occurrence of a string in the left-right order must be deleted (such an element is thought of having been copied)

Following these conventions, the interpretative result at the root of the previous tree, which is:

every linguist every linguist (some language) speaks /some language/

gives us two interpretative forms:

- the phonetic form:

  /every linguist//speaks//some language/

- the logical form:

  (every linguist)(some language)(speaks)

5.2 SOV languages

The SOV word order is obtained by making a verb a strong case assigner (exactly like [9]). In this case, the whole phrase some language is copied when using the product- elimination rule $[\otimes E]$, thus giving:

```
every linguist some language speaks
```

yielding:

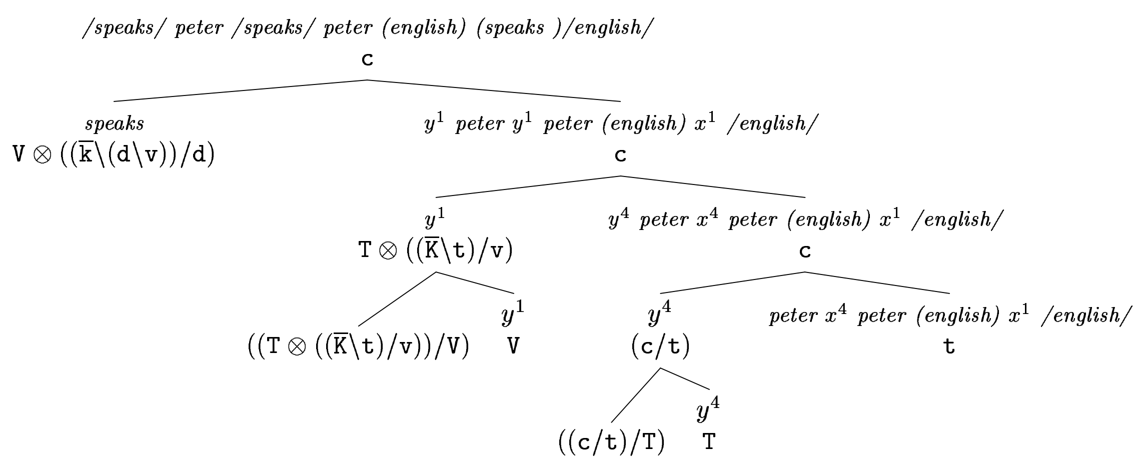
- phonetic form: */every linguist//some language//speaks/*
- logical form: *(every linguist)(some language)(speaks)*

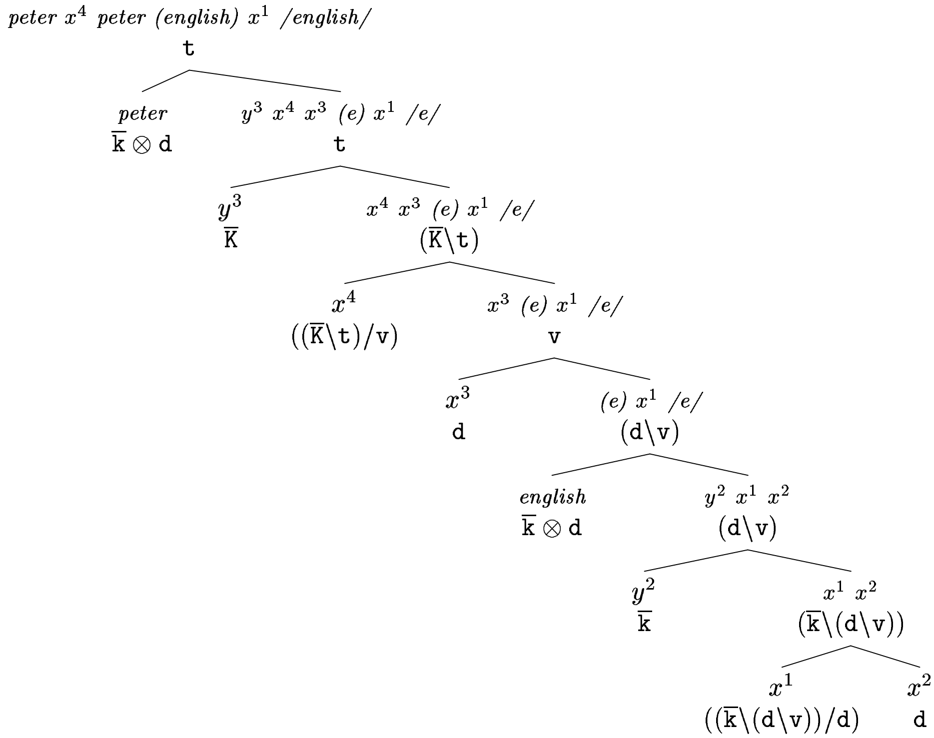
5.3 VSO languages

VSO languages are more difficult to obtain because they involve head movement. Head movement implies that categorial features too can be strong. In this case, in our setting, we duplicate the result category of a functorial type, combining the functorial type with this duplication of the result category seen as a strong feature by means of $\otimes$. The following lexicon gives an example (the resulting v is duplicated as V in the verbal entry, in order to correspond with the strong V demanded by the inflection category). The translation of Stabler's lexical entries is therefore the following:

entry	*Stabler'stype*	*label : type*
Peter	$\mathtt{d}\ -\overline{\mathtt{k}}\ peter$	$peter : \overline{\mathtt{k}} \otimes \mathtt{d}$
english	$\mathtt{d}\ -\overline{\mathtt{k}}\ english$	$english : \overline{\mathtt{k}} \otimes \mathtt{d}$
speaks	$=\mathtt{d}\ +\overline{\mathtt{k}}\ =\mathtt{d}\ \mathtt{v}\ speaks$	$speaks : \mathtt{V} \otimes ((\overline{\mathtt{k}}\backslash(\mathtt{d}\backslash\mathtt{v}))/\mathtt{d})$
(tense)	$=\mathtt{V}\ +\overline{\mathtt{K}}\ \mathtt{t}$	$((\mathtt{T} \otimes ((\overline{\mathtt{K}}\backslash\mathtt{t})/\mathtt{v}))/\mathtt{V})$
(comp)	$=\mathtt{T}\ \mathtt{c}$	$((\mathtt{c}/\mathtt{t})/\mathtt{T})$

An example of derivation :(we represent this tree in two parts)





In the result, repetitions are omitted, thus producing:

```
/speaks//peter/(peter)(english)(speaks)/english/
```

thus providing the following PF and LF:

- */speaks peter english/*
- *(peter)(english)(speaks)*

6 Conclusion

We present here a very simple logical system which does the same job as minimalist grammars do. It is a kind of intuitionistic linear logic where two products are mixed: one is non commutative and is not directly used, it is indirectly used via its residuates / and \, the other is commutative and is directly used for discharging hypotheses corresponding to two different insertion points in a tree (T-marker) where some definite material may occur. A simple algorithm is used afterwards to clean up the result thus making phonetic forms and logical forms appear. It may be shown that such a system has the same generative power as minimalist grammars (which have been proved to be *mildly context-sensitive*[5]), and that it is polynomial [8].

References

[1] N. Chomsky. Minimalist inquiries: the framework. Technical report, MIT, Cambridge, 1998.

[2] T. Cornell. A type-logical perspective on minimalist derivations. In G. van Kruijff and R. Oehrle, editors, *Formal Grammar'97*, Aix-en-Provence, 1997. ESSLLI'97.

[3] P. de Groote. Partially commutative linear logic: sequent calculus and phase semantics. In M. Abrusci and C. Casadio, editors, *Proofs and Linguistic Categories*, pages 199–208. CLUEB - University of Chieti, 1996.

[4] A. Lecomte. Categorial minimalism. In Moortgat [7], pages 143–158.

[5] J. Michaelis. Deriational minimalism is mildly context sensitive. In Moortgat [7], pages 179–198.

[6] M. Moortgat. Categorial type logics. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, chapter 2, pages 93–178. Elsevier, 1997.

[7] M. Moortgat, editor. *Logical Aspects of Computational Linguistics*, volume 2014 of *LNCS/LNAI*. Springer, 2001.

[8] C. Retoré. Polynomiality of pcll limited to application rules. Technical report, IRISA, Rennes, forthcoming.

[9] E. Stabler. Derivational minimalism. In C. Retoré, editor, *Logical Aspects of Computational Linguistics*, volume 1328 of *LNCS/LNAI*, pages 68–95. Springer, 1997.

[10] E. Stabler. Remnant movement and complexity. Technical report, UCLA, Los Angeles, 1999.

[11] W. Vermaat. Controlling movement: Minimalism in a deductive perspective. Technical report, OTS, Utrecht, 1999.