
Semantics in Minimalist-Categorial Grammars

ALAIN LECOMTE

Abstract

This paper is an attempt to develop a strictly derivationalist version of Chomsky's Minimalist theory integrating θ -roles and quantifier scoping without referring to a LF level. It is assumed different readings of a same sentence are obtained by various evaluation strategies, in a calculus which integrates a non deterministic version of $\lambda\mu$ -calculus.

Keywords SCOPE AMBIGUITIES, THETA-ROLES, MINIMALIST GRAMMARS, CATEGORIAL GRAMMARS, $\lambda\mu$ -CALCULUS

5.1 Introduction

Many works have been done on Type-Theoretical Grammars since the famous books by Glyn Morrill (Morrill (1994)) and Aarne Ranta (Ranta (1994)), respectively based on the Categorial tradition (Lambek (1958), Moortgat (1997)) and on Martin-Löf's Constructive Type Theory. More recently, much has been done, exploiting Curry's distinction between the tectogrammatical and the phenogrammatical levels, and this has led to interesting proposals like Lambda Grammars, Abstract Categorial Grammars and Convergent Grammars (de Groote (2001a), Muskens (2003), Pollard (2007)). Type-theoretic formulations of Minimalist Grammars have also been proposed (Lecomte and Retoré (2001), Amblard (2007), Lecomte (2005), Anoun and Lecomte (2006)). All these works take care of problems like scope ambiguities which are traditional in the Montagovian perspective but they pay little attention to thematic roles and binding phenomena (except Anoun and Lecomte (2006) and Pollard (2007)). These questions have been more widely adressed in the Generative frameworks, but unfortunately without giving a proper and

rigorous account of the derivations and above all of the syntax-semantics interface. In Chomsky's Minimalist Program, we can say that little attention is given to the conceptual structure (contrarily, say, to Jackendoff). Logical Form is simply a grammatical level, which remains very poor with regards to the interpretation. Moreover, if it is simply a level of Universal Grammar, the question arises whether such an extra level is really needed. Some, like C. Pollard (Pollard (2007)) have suggested that LF is mainly a way to take scope ambiguity into account, by means of *ad hoc* transformations of *Quantifier Raising*, the only displacements which occur after *Spell Out*. It is therefore tempting to develop a frame which keeps rigorous aspects of Categorical Grammar and reconciles them with some intuitions of Generativist linguists about the thematic structure, in order to get richer semantic representations through linguistic derivation.

In a nutshell, our proposal consists in using a bi-dimensional calculus, one dimension devoted to (narrow)-syntax, and the other to semantics (or "logical form" but in a more elaborated version than it is the case in Minimalism). In this sense, it has several points in common with Pollard (2007) and Pollard (2008) which recommends the syntax-semantics interface be *purely derivational* and *parallel*. By *purely derivational* he means that derivations are *proofs*, and by *parallel* that there are separate proofs that provide, respectively, candidate syntactic and semantic proofs and that it is the object of linguistic theory to specify those proof pairs that belong to the language in question.

Here, our viewpoint is slightly different: like in the traditional type-theoretic formalisms which, following Montague, are in favor of a *functional* approach to semantic interpretation (along the lines of the Curry-Howard correspondance), we assume that the (narrow) syntactic derivation functionally provides a semantic form, BUT this form is *underspecified*, that is: it may give various readings according to the way it is evaluated. This evaluation is performed according to various strategies, which are known in the theoretical computer science literature under the names of Call-by-value and Call-by-name (or variants) and it consists in *normalisation* procedure which are applied *after* the syntactic part of the calculus. This consists in fact in *switching* from the syntactic proof to the semantic one (by means of the translation from syntactic types to semantic ones) and then normalising the semantic proof. The point is that in those semantic proofs, a particular type (t) may be interpreted as the formula \perp , thus introducing into the calculus *negation* and rules for introducing and eliminating it: this justifies $\lambda\mu$ -calculus, since we know that it gives a computational content to *classical* logic (a logic where negation exists and where double negation may be eliminated).

Moreover, the results of evaluation are not predicate (or intensional) logic formulae *à la* Montague, but (fragments of) Discourse Representation Struc-

tures, simply because questions of binding are easier to solve in such a framework than in predicate (or intensional) logic. Pieces of information on the same entity are given at various places during a proof. Sometimes it could appear that a variable is bound before a new information is provided (like in donkey sentences), thus failing in the attribution of this information (which can be for instance an information on the thematic role). In such a case, the intermediate level of *discourse markers* reveals useful.

We must also add that the syntactic machinery is here provided by a piece of logical calculus (the so called *mixed-calculus*, invented by Philippe de Groote (de Groote (1996)) and worked out by Christian Retoré and Maxime Amblard (Amblard and Retoré (2007)), a fragment which has been proven equivalent to Stabler's minimalist grammars (Amblard (2007)). When translated into the sequent calculus, we are only using *cut-free proofs*. Because of that, the fact that we confine ourselves in this fragment has no severe consequences. Of course the use of the cut-rule and of the cut-elimination procedure would lead us to get off this fragment, thus obtaining more proofs, some of them having perhaps no linguistic interpretation. Another alternative is to keep Minimalist Grammars as they are, using them as mere guidelines for obtaining semantic proofs that could still be normalized afterwards like we do here.

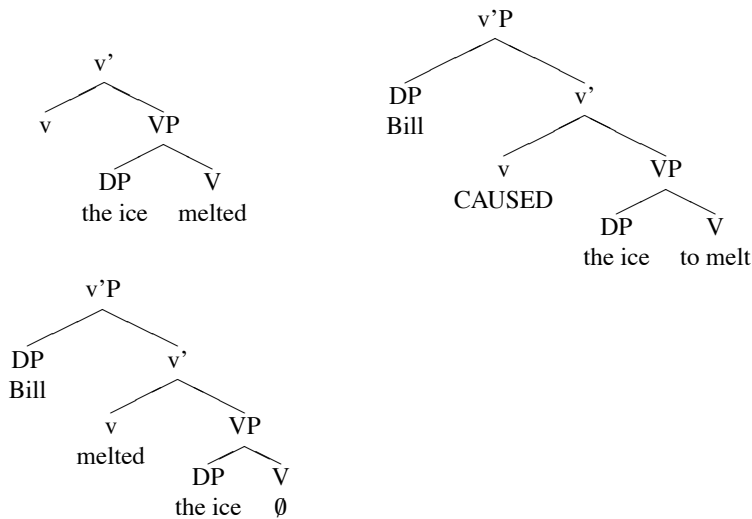
5.2 Elements of VP analysis

Various works (Davidson (1966), Vendler (1967)), have led to the idea that verbs express events and that there are complex events, which are structured into an *inner* and an *outer* event, where the outer one is associated with causation and agency, and the inner one is associated with telicity and change of state. This semantic idea is reflected in syntax by the introduction of a *v* label which is added to *V* into the structure of VP in order to distinguish between an internal part and an external one. Sometimes (see Hale and Keyser (1993)) it is said that *v* is associated with protoverbs like *DO* or *ACT*.

Such views make it possible to understand *transitivity alternations*, that is cases where a same verb can appear alternately in causative (*a*) or inchoative (*b*) sentences, like:

- Example 1** a. *Bill melted the ice*
 b. *the ice melted*

(a) may be rephrased as *Bill caused the ice to melt*, and the history of derivations may be represented as the three following trees.



where the first tree corresponds to (b), the third one to (a) and the second one to the rephrasing of (a). As we see, v can be either interpreted as a protoverb (introducing a cause) or as an empty node which can serve as a target for a movement from V .

In parallel, questions have been asked on how the semantic arguments of a verb are linked to syntactic positions, in such a way that for instance *agents* are generally *subjects* and *themes* are generally *objects*. This question led to the idea of an attribution order of the so-called θ -roles, which has been expressed by M. Baker (Baker (1997)) in the following principle, known as UTAH (*Uniform Theta Assignment Hypothesis*):

Two arguments which fulfill the same thematic function with respect to a given predicate must occupy the same underlying position in the syntax.

If we put together these ideas we are led to a structure of VP with *different levels*, such that AGENTS are introduced in a specifier position of the highest VP (an "external" position), whereas THEMES are introduced in a specifier position of the lowest (an "internal" one).

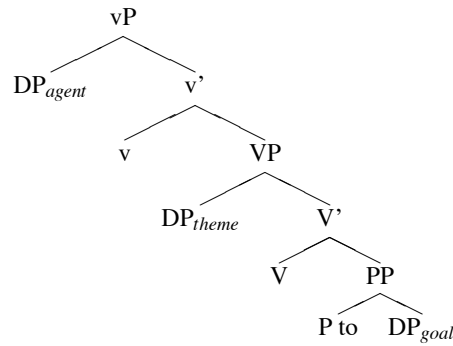
Moreover these principles help us finding a structure for ditransitive (or *double-object*) constructions like :

Example 2 John gave a book to Mary

and for structures with *resultative* predicates, like:

Example 3 The acid will turn the paper red

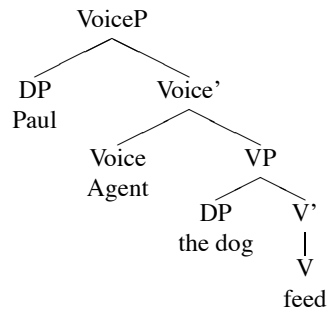
From these observations, the canonical structure of a VP has been defined as:



If Hale & Keyser conceive v as proto-verbs expressing action or causation, other researchers have assumed that agents are introduced by a special predicate, which is independent and additional to a transitive VP and that this predicate selects an agent and the event described by the verb. Many considerations show that this predicate corresponds to an inflectional head. Kratzer (Kratzer (1994)) assumes this head to be *Voice*. In this paper, we shall concentrate on this version of so-called "constructionalism".

In all the sequel, e will denote the type of *individuals* and s the type of *events*.

According to Kratzer (Kratzer (1994)), External Arguments are base-generated in SPEC of VoiceP, and direct objects in SPEC of VP, thus leading to the following analysis of *Paul feeds the dog*:



The semantic interpretation of which is provided by the following steps:

1. **feed*** = $\lambda x_e \lambda e_s. feed(x)(e)$
2. **the dog*** = the dog
3. **(the dog feed)*** = $\lambda e_s. feed(the\ dog)(e)$
4. **Agent*** = $\lambda x_e \lambda e_s. Agent(x)(e)$

$$5. \text{ (Agent (the dog feed))}^* = \lambda x_e \lambda e_s. \text{Agent}(x)(e) \wedge \text{feed}(\text{the dog})(e)$$

$$6. \text{ Paul}^* = \text{Paul}$$

$$7. \text{ (Agent (the dog feed)) Paul}^* = \lambda e_s. \text{Agent}(\text{Paul})(e) \wedge \text{feed}(\text{the dog})(e)$$

where the step (5) rests on a *ad hoc* rule that Kratzer names *Event Identification* :

It takes a function f and a function g as input and yields a function h as output. f is of type $\langle e, \langle s, t \rangle \rangle$ and g of type $\langle s, t \rangle$. h is of type $\langle e, \langle s, t \rangle \rangle$. The function h is calculated by the rule: $h = \lambda x_e \lambda e_s. f(x)(e) \wedge g(e)$

One of the goals of this paper is to provide a solution which avoids such operations. This solution is based on a Type calculus, that we name the *Categorial-Minimalist framework* (Amblard (2007), Lecomte and Retoré (2001), Lecomte (2005)). Moreover, we keep in mind that we don't only seek a solution for these verbal puzzles, but we also still want to express the usual properties of determiner phrases (and in particular of *Generalized Quantifiers*) along lines similar to the *Categorial framework* (Moortgat (1997)).

5.3 The mixed calculus and the categorial-minimalist framework

5.3.1 A Labelled Type Grammar

Our framework is based on the mixed calculus (de Groot (1996), Amblard and Retoré (2007)), a formulation of Partially Commutative Linear Logic. The plain calculus contains introduction and elimination rules for the non commutative product \bullet and its residuals $/$ and \backslash , and for the commutative product \otimes and its residual $-\circ$. Moreover there is an entropy rule which allows to relax the order between hypotheses. This calculus has been shown to be normalizable (Amblard and Retoré (2007)). For its use in linguistics however, we restrict it to elimination rules of $/$, \backslash and \otimes since we don't see particular evidence for using introduction rules (there are hypotheses in this calculus but they are discharged by means of the \otimes elimination rule).¹ The operations *Merge* and *Move* of Minimalist Grammars (Stabler (1997), Stabler (2001)) are replaced and simulated by combinations of logical rules, labelled with strings for the phonological parts and by $\lambda\mu$ -terms for the semantics (Amblard

¹We therefore accept the critics that it is not a "Type Logic", it is the reason why we speak rather of a "Type Grammar"

(2007)). *Merge* is elimination of / or \ followed by *entropy*:

$$\frac{\frac{\Delta \vdash u : A \quad \Gamma \vdash w : A \setminus C}{\Delta; \Gamma \vdash uw : C} [\setminus E]}{\Delta, \Gamma \vdash uw : C} [entropy] \quad \frac{\frac{\Gamma \vdash w : C/A \quad \Delta \vdash u : A}{\Gamma; \Delta \vdash wu : C} [/E]}{\Gamma, \Delta \vdash wu : C} [entropy]$$

Move is $[\otimes E]$.

$$\frac{\Gamma \vdash (z_1, z_2) : A \otimes B \quad \Delta, x : A, y : B, \Delta' \vdash t : C}{\Delta, \Gamma, \Delta' \vdash t[z_1/x, z_2/y] : C} [\otimes E]$$

The comma (",") is the structural counterpart of the commutative product, while the semi-column (";") is the counterpart of the non-commutative product.

5.3.2 The semantical tier

Semantical representations are built in by means of $\lambda\mu$ -terms. Let us simply recall here that the $\lambda\mu$ -calculus is a strict extension of the λ -calculus (Parigot (1992), de Groote (2001b)). Its syntax is provided with a second alphabet of variables ($\alpha, \beta, \gamma, \dots$ - called μ -variables), and two additional constructs: μ -abstraction ($\mu\alpha.t$), and naming (αt). The syntax of Parigot's $\lambda\mu$ -calculus is (Parigot (1992)):

$$\begin{aligned} V &::= x | \lambda x.v \\ v &::= V | (v v) | \mu\alpha.c \\ c &::= (\alpha v) \end{aligned}$$

where V 's are called *values*. We shall see later on the role these *values* play in the choice of an evaluation strategy.

μ -reduction may be defined in two ways, according to two rules that we shall denote by (μ) and (μ') .

$$(\mu) \quad (\mu\alpha.u)v \longrightarrow \mu\beta.u[\alpha t := \beta(t v)]$$

where $u[\alpha t := \beta(t v)]$ stands for the term u where each subterm of the form (αt) has been replaced by $\beta(t v)$.

$$(\mu') \quad v(\mu\alpha.u) \longrightarrow \mu\beta.u[\alpha t := \beta(v t)]$$

μ -reductions take their contexts (or "continuations") as their arguments and put them inside their body at the points which are marked by the corresponding μ -variable. With these two rules the calculus is not deterministic (see again de Groote (2001b)), but this is exactly what we want when we wish to have an account, say, of scope ambiguities.

In order to end up a reduction, we shall assume moreover one simplification rule which applies when there is no more continuation to pass:

$$(\sigma) \quad \mu\alpha.u \longrightarrow u[\alpha t := t]$$

The problem which arises when we deal with the assignment of theta roles is that the derivation of a VP is such that meaning components concerning the event to which a verb refers are given at different stages. For instance, according to the principle that the THEME or PATIENT role is assigned in a *specifier* position with regards to the lowest part of the verb, it is assumed that the corresponding argument is raised from its *complement* position (where it feeds the verbal entry with a suitable variable or *reference marker*) to the specifier position (where it receives its θ -role).

Because movement is expressed by the $[\otimes E]$ rule, that mechanism supposes that two variables, each associated with a hypothesized type (e.g. d and k) are discharged at the same time by the respective components of a same pair (e.g. associated with a $d \otimes k$ type), but these two components are relative to the very same object, that we propose to consider as a same *reference marker*. This leads us to use a formalism which is very close to Kamp's *DRSs*.

We will then actually use DRSs as semantic recipes (see for instance R. Muskens and Visser (1997)), a way of getting a dynamic system which allows us to merge several *conditions* on a same *referent* at different times. We shall therefore have *discourse referents* and *variables*, the first ones will be noted with a point on their top ($\dot{x}, \dot{y}, \dot{z}, \dots$). A discourse referent is an individual which can be passed as a value to any individual variable.

The syntax of structures is provided by:

$$\begin{aligned} \gamma & ::= (P \dot{\xi}) | (\alpha \dot{\xi}) | \xi_1 = \xi_2 | \neg K | K_1 \vee K_2 | K_1 \Rightarrow K_2 \\ K & ::= [\xi_1 \dots \xi_n | \gamma_1, \dots, \gamma_m] \end{aligned}$$

where each ξ or ξ_i is either a variable or a referent. P is a *predicate-variable*. α is a μ -variable (also called a *co-variable*). λ -abstraction operates only on variables (individual or predicate), and μ -abstraction on co-variables.

An operation on pairs of DRSs or on pairs (*DRS*, *condition*) is *Fusion* (elsewhere named *Merge*, but we reserve this naming to the syntactic operation), that we will denote \sqcup . We don't develop this point here (see Zeevat (1991), Vermeulen (1993) and more recently de Groote (2007), van Eijck (2007), Muskens (1996)) because it is not the object of the present paper. It suffices to know that we have at least two alternatives for defining *Fusion*:

1. $[V|F] \sqcup_1 [W|G] = [V \cup W | F \cup G]$
2. $[V|F] \sqcup_2 [W|G] = [V \oplus W | F^* \cup G^*]$

In (1) the same discourse referents which occur in V and in W are identified. In (2), they are renamed before fusion (and F^* and G^* are the conditions in F and G taking this renaming into account). If the choice between these two modes is sometimes ambiguous when dealing with discourse, that will not be the case in the simple frame of one sentence.

Nevertheless, this is not enough for defining *Fusion*, because in most of

our examples we shall have to make the fusion of conditions containing an arbitrary set of variables with complex DRSs where the same variables (or a subset of these) are introduced at different levels. That leads to a more complex definition.

Definition 1 Let D be a complex DRS, let \prec the relation of subordination between sub-DRSs and DRSs, let R_D the set of reference markers of D , let m_δ the set of reference markers accessible from the sub-DRS δ , let M_δ the set of reference markers contained inside δ , let C_δ the set of conditions inside δ (some of which being sub-DRSs). Let $[V|\phi]$ a DRS expressing a single condition on variables in V . Let us suppose that $V = V_1 \cup V_2$ with $V_2 \cap M_D = \emptyset$ and $V_1 \subset M_D$.

- if $V_1 \subset R_D$ then $D \sqcup [V|\phi] = [R_D \cup V_2 | C_D \cup \{\phi\}]$
- else if $\exists \delta \prec D$ and $V_1 \subset m_\delta$
 - if $\delta = \delta_1 \Rightarrow \delta_2$ then $D \sqcup [V|\phi] = [\delta_2 \leftarrow [R_\delta \cup V_2 | C_\delta \cup \{\phi\}]]D$
 - else, $D \sqcup [V|\phi] = [\delta \leftarrow [R_\delta \cup V_2 | C_\delta \cup \{\phi\}]]D$ where " \leftarrow " expresses the substitution.

The terms of our calculus are therefore the previous structures and those which are obtained by λ or μ -abstraction on them, like for instance:

$$\lambda Q. \mu \alpha. [\dot{x} | (Q \dot{x}) \wedge (\alpha \dot{x})]$$

Let us take for instance a μ -term like (1) $\mu \alpha. [\dot{x} | (BOOK \dot{x}) \wedge (\alpha \dot{x})]$.

If we admit $[\dot{x} | (BOOK \dot{x}) \wedge (\alpha \dot{x})]$ is of type t (because it represents the Discourse Structure of a sentence), $\mu \alpha$ makes at a same time an abstraction on α (thus leading to the type $(e \rightarrow t) \rightarrow t$) and performs the elimination of double negation, thus leading finally to the type e . Therefore, when we have a semantic recipe like (1), it is of type e and we are allowed to put it as an e -type argument of a function like a transitive verb (of type $e \rightarrow e \rightarrow t$). We prove this typing by the following derivation (where the distinguished type t is considered the "observable" type which can also be seen as \perp , and therefore, $e \rightarrow t$ is equivalent to $e \rightarrow \perp$ and equivalent to $\neg e$):

$$\frac{\frac{\Gamma \vdash BOOK : e \rightarrow \perp \quad \Gamma \vdash \dot{x} : e \quad \alpha : \neg e \vdash \alpha : e \rightarrow \perp \quad \Gamma \vdash \dot{x} : e}{\Gamma \vdash BOOK(\dot{x}) : \perp \quad \Gamma, \alpha : \neg e \vdash (\alpha \dot{x}) : \perp}}{\Gamma, \alpha : \neg e \vdash [\dot{x} | BOOK(\dot{x}) \wedge (\alpha \dot{x})] : \perp} [E \perp]}{\Gamma \vdash \mu \alpha. [\dot{x} | BOOK(\dot{x}) \wedge (\alpha \dot{x})] : e}$$

Let us notice that \dot{x} , as a discourse referent, is introduced here like a constant, it is exactly as if the context (here represented by Γ) provided a new marker each time it is needed.

Such properties are important because they will allow us to state that:

- scope is no longer dependent on *c-command* (a binder may bind the rest of a sentence even if remaining *in situ*)
- quantified expressions no longer need higher order types (they can take their scopes from inside the terms in which they are enclosed, keeping their original type e for an *NP*, for instance)

Another $\lambda\mu$ -term for our grammar is:

$$\lambda Q.\mu\alpha.[|\dot{x}|(Q \dot{x})] \Rightarrow [|\alpha \dot{x}|]$$

which is introduced to serve as the translation of *every*, like in

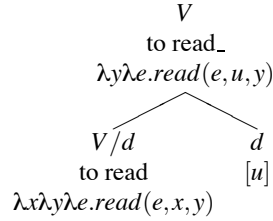
every child reads a book

The term $\mu\alpha.[|\dot{x}|(Q \dot{x})] \Rightarrow [|\alpha \dot{x}|]$ is also of type e by the same reasoning.

5.4 Application to the VP structure

5.4.1 VP syntactic derivation

Let us assume a verb like *to read* has the syntactic type V/d , the semantic type $e \rightarrow e \rightarrow s \rightarrow t$ and the semantic recipe $\lambda x\lambda y\lambda e.read(e, x, y)$. In a first step of the derivation, it will receive a first argument by merging with a hypothesis u of type d (hypotheses are put inside square brackets).



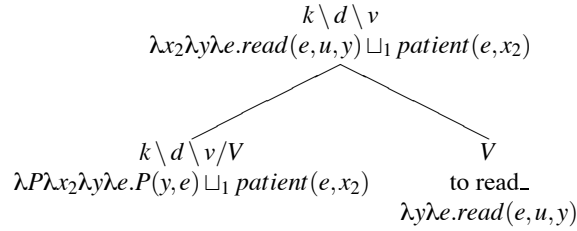
We then assume a *phonologically empty* type corresponding to the adding of the first thematic role (*patient*), of syntactic type $k \setminus d \setminus v/V$ and semantic recipe:

$$\lambda P\lambda x_2\lambda y\lambda e.P(y, e) \sqcup_1 patient(e, x_2)$$

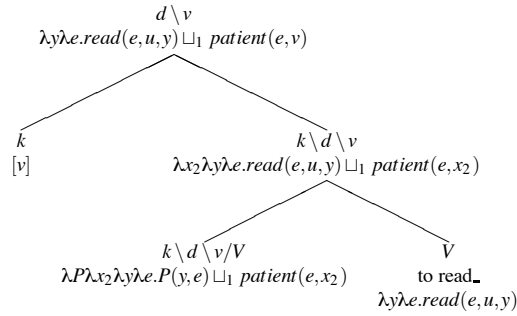
The fusion operation \sqcup_1 is introduced at this point even if it is not particularly relevant because we have still only atomic formulae and neither DRS nor "conditions", but it is not difficult to define an extension of Fusion to these formulae: it simply amounts to conjunction in this particular case.

By *merge* with the previously obtained V , we get:

$$\lambda x_2\lambda y\lambda e.read(e, u, y) \sqcup_1 patient(e, x_2)$$



At this point, a second hypothesis, v , of type k is introduced, so that it results in:



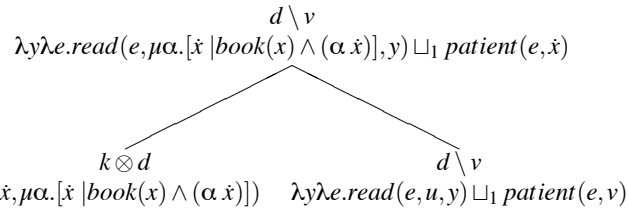
We can comment these steps by saying that:

1. the verbal phrase is prepared to host a first thematic role (patient)
2. the thematic role will be assigned to the *DP* which will raise towards the specifier position, here marked by the attribution of a formal feature k

The two hypotheses u and v can be discharged altogether by the $[\otimes E]$ rule, by means of the DP *a book*.

We assume this DP is of syntactic type $k \otimes d$, and of semantic type $e \otimes e$. Its semantic recipe is the pair $(\dot{x}, \mu\alpha.[\dot{x} \mid \text{book}(x) \wedge (\alpha \dot{x})])$.

This step, based on $[\otimes E]$, builds up the following tree:

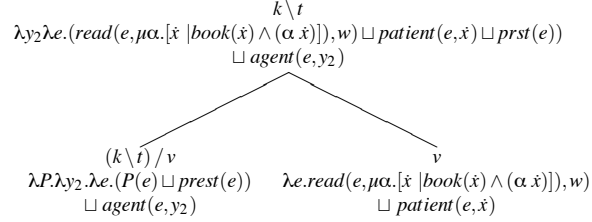


We get $\lambda y \lambda e . \text{read}(e, \mu\alpha.[\dot{x} \mid \text{book}(\dot{x}) \wedge (\alpha \dot{x})], y) \sqcup_1 \text{patient}(e, \dot{x})$, of syntactic type $d \setminus c$.

After this sequence of steps, there is a new merge, with a new hypothesis w of type d , thus giving $\lambda e . \text{read}(e, \mu\alpha.[\dot{x} \mid \text{book}(\dot{x}) \wedge (\alpha \dot{x})], w) \sqcup_1 \text{patient}(e, \dot{x})$

of type v .

Tense is then added by merge with an entry of type $(k \setminus t) / v$ which at the same time provides an agentive role the semantics of which is: $\lambda P.\lambda y_2.\lambda e.(P(e) \sqcup \text{prst}(e)) \sqcup \text{agent}(e, y_2)$, thus leading to:



This is merged again with a hypothesis z of type k thus providing the sequent:

$$\begin{array}{c}
 \Gamma, z : k, w : d \vdash \\
 t : \lambda e. ((\text{read}(e, \mu \alpha. [\dot{x} \mid \text{book}(\dot{x}) \wedge (\alpha \dot{x})]), w) \\
 \sqcup \\
 \text{patient}(e, \dot{x}) \sqcup \text{prst}(e)) \sqcup \text{agent}(e, z)
 \end{array}$$

When these hypotheses are discharged by a DP like:

$$(\dot{y}, \mu \beta. [[\dot{y} \mid \text{child}(\dot{y})] \Rightarrow [[(\beta \dot{y})]]])$$

we finally get

$$\begin{array}{c}
 \Gamma \vdash \\
 t : \lambda e. (\text{read}(e, \mu \alpha. [\dot{x} \mid \text{book}(\dot{x}) \wedge (\alpha \dot{x})]), \mu \beta. [[\dot{y} \mid \text{child}(\dot{y})] \Rightarrow [[(\beta \dot{y})]]]) \\
 \sqcup \\
 \text{patient}(e, \dot{x}) \sqcup \text{prst}(e) \sqcup \text{agent}(e, \dot{y})
 \end{array}$$

The final type c/t the semantic recipe of which is (for simplification) $\lambda P.P(\dot{e})$, where \dot{e} is a discourse referent of type s (event), finally gives the following semantic representation :

$$\begin{array}{c}
 \text{read}(\dot{e}, \mu \alpha. [\dot{x} \mid \text{book}(\dot{x}) \wedge (\alpha \dot{x})]), \mu \beta. [[\dot{y} \mid \text{child}(\dot{y})] \Rightarrow [[(\beta \dot{y})]]]) \\
 \sqcup \\
 \text{pt}(\dot{e}, \dot{x}) \sqcup \text{prst}(\dot{e}) \sqcup \text{agt}(\dot{e}, \dot{y})
 \end{array}$$

which does not yet provide an interpretable meaning. For that, steps of evaluation are still needed. It is the object of the next subsection. But we may summarize the previous discussion by giving the list of grammatical expressions that this derivation required.

| syntactic type | semantic recipe |
|---|---|
| V / d transitive verb | $\lambda x \lambda y \lambda e. read(e, x, y)$ to read |
| $k \setminus d \setminus v / V$ θ -role 1 | $\lambda P \lambda x_2 \lambda y \lambda e. P(y, e) \sqcup patient(e, x_2)$ ε |
| $k \setminus t / v$ θ -role 2 | $\lambda P \lambda x_2 \lambda e. (P(e) \sqcup prest(e)) \sqcup agent(e, x_2)$ Tense |
| $k \otimes d$ DP | $(\dot{x}, \mu\alpha. [\dot{x} book(\dot{x}) \wedge (\alpha \dot{x})])$ a book |
| $k \otimes d$ DP | $(\dot{x}, \mu\alpha. [[\dot{x} child(\dot{x}) \Rightarrow (\alpha \dot{x})]]])$ every child |
| c / t comp | $\lambda P. P(\dot{e})$ ε |

5.4.2 VP semantic evaluation

We will assume conditions $C(\dot{x}_1, \dots, \dot{x}_n)$ equivalent to DRSs :

$$[\dot{x}_1, \dots, \dot{x}_n | C(\dot{x}_1, \dots, \dot{x}_n)]$$

The substructure :

$$read(\dot{e}, \mu\alpha. [\dot{x} | book(\dot{x}) \wedge (\alpha \dot{x})], \mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]])$$

can be evaluated in two ways.

First calculation:

1. by μ' -reduction:

$(\alpha \dot{x})$ replaced by $(\alpha (read(\dot{e}) \dot{x}))$

$$\begin{aligned} & ((read(\dot{e}), \mu\alpha. [\dot{x} | book(\dot{x}) \wedge (\alpha \dot{x})]), \mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]) \\ & \longrightarrow \\ & (\mu\alpha. [\dot{x} | book(\dot{x}) \wedge (\alpha (read(\dot{e}) \dot{x}))], \mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]) \end{aligned}$$

2. by μ -reduction:

$(\alpha (read(\dot{e}) \dot{x}))$ replaced by $(\alpha ((read(\dot{e}) \dot{x}) (\mu\beta. [| [\dot{y} | \dots]]])))$

$$\begin{aligned} & (\mu\alpha. [\dot{x} | book(\dot{x}) \wedge (\alpha (read(\dot{e}) \dot{x}))], \mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]) \\ & \longrightarrow \\ & (\mu\alpha. [\dot{x} | book(\dot{x}) \wedge (\alpha ((read(\dot{e}) \dot{x}) (\mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]))]) \end{aligned}$$

3. by simplification rule σ

$$\begin{aligned} & (\mu\alpha. [\dot{x} | book(\dot{x}) \wedge \\ & (\alpha ((read(\dot{e}) \dot{x}) (\mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]))]) \\ & \longrightarrow \\ & [\dot{x} | book(\dot{x}) \wedge \\ & ((read(\dot{e}) \dot{x}) (\mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]))] \end{aligned}$$

4. by μ' -reduction:

$(\beta \dot{y})$ replaced by $(\beta ((read(\dot{e}) \dot{x}) \dot{y}))$

$$\begin{aligned} & [\dot{x} | book(\dot{x}) \wedge ((read(\dot{e}) \dot{x}) (\mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta \dot{y})]]]]))] \\ & \longrightarrow \\ & [\dot{x} | book(\dot{x}) \wedge (\mu\beta. [| [\dot{y} | child(\dot{y}) \Rightarrow [| (\beta ((read(\dot{e}) \dot{x}) \dot{y}))]]]]))] \end{aligned}$$

5. by simplification again

$$\begin{array}{c} [\dot{x} | \text{book}(\dot{x}) \wedge (\mu\beta. [| [\dot{y} | \text{child}(\dot{y}) \Rightarrow [| (\beta ((\text{read}(\dot{e}) \dot{x}) \dot{y}))]]]]]] \\ \longrightarrow \\ [\dot{x} | \text{book}(\dot{x}) [| [\dot{y} | \text{child}(\dot{y}) \Rightarrow [| (\text{read}(\dot{e}) \dot{x}) \dot{y})]]]]] \end{array}$$

This reading of course corresponds to the one in which at some instant there is a book which is read by every child.

In the second calculation, after (μ') , (μ') is still used, replacing $(\beta \dot{y})$ by $(\beta (\mu\alpha. [\dot{x} | \dots]))$, followed by simplification, (μ) and again simplification, leading to:

$$[| [\dot{y} | \text{child}(\dot{y}) \Rightarrow [| [\dot{x} | \text{book}(\dot{x}) \wedge ((\text{read}(\dot{e}) \dot{x}) \dot{y}))]]]]]$$

This reading corresponds to the one in which for every child, there is a book which is read by him or her on a particular event \dot{e} .

Now, if we take these two readings, we may combine them with the other parts of the resulting structure since we just obtained, on either side, a DRS (or a condition in a DRS). We obtain both structures:

$$\begin{array}{l} [\dot{e}, \dot{x} | \text{prst}(\dot{e}) \wedge \text{book}(\dot{x}) \wedge \text{pt}(\dot{e}, \dot{x}), \\ [| [\dot{y} | \text{child}(\dot{y}) \Rightarrow [| (\text{read}(\dot{e}) \dot{x}) \dot{y}) \wedge \text{agt}(\dot{e}, \dot{y})]]]]] \text{ and} \\ [\dot{e} | \text{prst}(\dot{e}), [\dot{y} | \text{child}(\dot{y}) \Rightarrow \\ [| [\dot{x} | \text{book}(\dot{x}) \wedge \text{pt}(\dot{e}, \dot{x}) \wedge ((\text{read}(\dot{e}) \dot{x}) \dot{y}) \wedge \text{agt}(\dot{e}, \dot{y}))]]]]] \end{array}$$

where *Fusion* is defined like in (5-3).

5.5 Binding

One of the advantages of this framework is in its ability to easily deal with binding phenomena. Let us imagine the following supplementary lexical entries.

| syntactic type | semantic recipe |
|---|--|
| V / d stative verb | $\lambda x \lambda e. \text{smart}(e, x)$ to be smart |
| V / c prop verb | $\lambda x \lambda y \lambda e. \text{think}(e, x, y)$ to think |
| $k \setminus t / V$ θ -role 1 | $\lambda P \lambda x_2 \lambda e. (P(e) \sqcup \text{state}(e)) \sqcup \text{expc}(e, x_2)$ Tense |
| $k \setminus d \setminus v / V$ θ -role 1 | $\lambda P \lambda x_2 \lambda y \lambda e. P(y, e) \sqcup \text{patient}(e, x_2)$ ε |
| $k \otimes d$ ana | $(\text{ana}(\dot{x}), \text{ana}(\dot{x}))$ (s)he |
| $k \otimes d$ refl | $(\text{refl}(\dot{x}), \text{refl}(\dot{x}))$ him/herself |
| c / t comp | $\lambda P. P(\dot{e})$ ε |

For the following sentences:

- Example 4** a. Peter thinks he is smart
b. John thinks Paul shaves himself

what we get is ²:

(4 a) :

$$[\dot{e}, \dot{y}] \dot{y} = Peter, agent(\dot{e}, \dot{y}), think(\dot{e}, [\dot{e}_1, \mathbf{ana}(\dot{x}) | smart(\dot{e}_1, \dot{x}), expc(\dot{e}_1, \dot{x})], \dot{y})]$$

(4 b):

$$[\dot{e}, \dot{y}] \dot{y} = John, agent(\dot{e}, \dot{y}), think(\dot{e}, [\dot{e}_1, \mathbf{refl}(\dot{x}), \dot{z} | \dot{z} = Paul, shave(\dot{e}_1, \dot{x}, \dot{z}), pt(\dot{e}_1, \dot{x}), agt(\dot{e}_1, \dot{z})], \dot{y})]$$

A sentence is said to be closed when its event variable has been instantiated (by an event-reference marker). It is assumed here that every closed sentence must have its **ana**- and **refl**- reference markers linked either inside the sentence (S-linked) or by the discourse (in which case they are D-linked). It is assumed that all **refl**- reference markers must be S-linked, and all **ana**-reference markers must be S-linked or D-linked.

It is otherwise assumed that a **ana**- or a **refl**- reference marker may be S-linked only by being identified with an ordinary reference marker (of the same type, *e*, *s* or *t*) in the same sub-DRS for **refl** and in the immediate higher DRS for **ana** (of course it may be linked to it by an identification chain). Then this provides convenient readings for both sentences: (4 a) :

$$[\dot{e}, \dot{y}] \dot{y} = Peter, agent(\dot{e}, \dot{y}), think(\dot{e}, [\dot{e}_1 | smart(\dot{e}_1, \dot{y}), expc(\dot{e}_1, \dot{y})], \dot{y})]$$

$$[\dot{e}, \dot{y}] \dot{y} = Peter, agt(\dot{e}, \dot{y}), think(\dot{e}, [\dot{e}_1, \dot{x}_D | smart(\dot{e}_1, \dot{x}), exp(\dot{e}_1, \dot{x})], \dot{y})]$$

(4 b):

$$[\dot{e}, \dot{y}] \dot{y} = John, agt(\dot{e}, \dot{y}), think(\dot{e}, [\dot{e}_1, \dot{z} | \dot{z} = Paul, shave(\dot{e}_1, \dot{z}, \dot{z}), pt(\dot{e}_1, \dot{z}), agt(\dot{e}_1, \dot{z})], \dot{y})]$$

We don't treat here the relations between events. The multi-occurrence of event-reference markers opens the field to the exploration of tense correspondances inside complex sentences.

5.6 Conclusion and perspectives

The previous analysis may pose some problems:

1. more readings exist for sentences including an event argument (for instance readings like *there is a book such that every child reads this book at some event* or *every child reads a book at some event*),

²**ana** and **refl** are essentially syntactic devices, on the semantic side, they can be interpreted as identity functors

2. not all readings are required in all cases: for instance in case of a question, the *wh*-DP may have always a wide scope over any QNP

The first problem is solved by giving to the empty *comp* the semantic recipe $\lambda P.P(\mu\delta.[\dot{e}|(\delta \dot{e})])$. The second problem is more delicate: it is here that the choice between several strategies is relevant. Semantic recipes associated with some *wh*-DP or QNP may be labelled by the kind of regime they ask for.

As we know from works on duality of computation (among them Herbelin (2005), Curien and Herbelin (2000)), several reduction systems may be considered, which are deterministic. Among the deterministic calculi, let us mention the *left to right call by value* and the *right to left call by value* versions.

Left to Right Call by Value

$$\begin{array}{lll}
 (\beta_v) & ((\lambda x.v) V) & \rightarrow v[x \leftarrow V] \\
 (\mu_L) & ((\mu\alpha.c) v) & \rightarrow \mu\alpha.c[\alpha \leftarrow (\alpha ([] v))] \\
 (\mu_{Rv}) & ((\lambda x.v) (\mu\alpha.c)) & \rightarrow \mu\alpha.c[\alpha \leftarrow (\alpha ((\lambda x.v) []))] \\
 (\mu_{var}) & (\alpha \mu\beta.c) & \rightarrow c[\beta \leftarrow (\alpha [])]
 \end{array}$$

Notice that in that version,

- the β reduction rule only applies to *values*, that is not, for instance, to μ -terms
- the (μ) reduction rule is kept free of any conditions
- the (μ') rule may be applied, but only in the context of a λ -term

Let us look at then the right to left call by value version.

Right to Left Call by Value

$$\begin{array}{lll}
 (\beta_v) & ((\lambda x.v) V) & \rightarrow v[x \leftarrow V] \\
 (\mu_{Lv}) & ((\mu\alpha.c) V) & \rightarrow \mu\alpha.c[\alpha \leftarrow (\alpha ([] V))] \\
 (\mu_R) & (v (\mu\alpha.c)) & \rightarrow \mu\alpha.c[\alpha \leftarrow (\alpha (v []))] \\
 (\mu_{var}) & (\alpha \mu\beta.c) & \rightarrow c[\beta \leftarrow (\alpha [])]
 \end{array}$$

Symmetrically with regards to the left to right version, we may observe that:

- the β reduction rule only applies to *values*, that is not, for instance, to μ -terms
- the (μ) reduction rule only applies when a μ -term is applied to a value
- the (μ') reduction rule is kept free of any conditions

Now, if we look at our two calculations in 4-2, we see that for each one, the second step is crucial, because there, expressions of the form $(\mu\alpha.\phi, \mu\beta.\psi)$ are met. Under a *LRCBV* regime, (μ') cannot be applied (thus blocking the second calculation) and under a *RLCBV* regime, it is (μ) which cannot be applied, thus blocking the first calculation. Therefore, if we wish to block the

reading where the existential *a book* is raising, we specify that $\mu\alpha$ is under the *RLCBV* regime, and we write $\mu^-\alpha$. In this case only the second reading will obtain. Reciprocally, if we wish to compel this existential to raise, we simply specify that $\mu\beta$ is under the *LRCBV* regime, and we write $\mu^+\beta$. This means that the complete evaluation will be held under this regime. By default, μ is used with an underspecified regime thus resulting in several readings. Notice that two different labels for μ operators in the same sentence would result in a conflict (a *crash* of the calculation).

Finally, we may compare this proposal with the ideas included in the Minimalist Program: whereas, in MP, a part of the derivation is continued after *Spell-Out*, which consists in a "covert" part of the derivation (mainly Quantifier-Raising), we propose to replace these transformations by an evaluation stage which is done according to a well-defined strategy.

Acknowledgements

Thanks to Ruth Kempson, Glyn Morrill and Carl Pollard for providing a helpful discussion during the workshop Glyn organized in Barcelona in December, thanks also to Maxime Amblard for his suggestions and advices, to Lea Nash having attracted my attention on "constructionnalism" and to Philippe de Groote, Michael Moortgat and Hugo Herbelin for having given to me a better understanding of the $\lambda\mu$ -calculus (and other related formalisms) by the talks they gave in the Prelude workshop held last year in Carry-le-Rouet and to three anonymous referees who helped much in formulating some parts of this paper. Of course, all of the incorrections are of my own responsibility.

References

- ACL'01. 2001. *Proceedings of the 39th Meeting of ACL*. Toulouse: ACL 2001.
- Amblard, M. 2007. *Calculs de représentations sémantiques et syntaxe générative: les grammaires minimalistes catégorielles*. Phd thesis, Université Bordeaux 1.
- Amblard, M. and C. Retoré. 2007. Natural deduction and normalization for partially commutative linear logic and Lambek calculus with product. *Computation and Logic in the Real World* Quaderni del Dipartimento di Scienze Matematiche Roberto Maggiori.
- Anoun, H. and A. Lecomte. 2006. Linear grammars with labels. In G. S. P. Monachesi, G. Penn and S. Wintner, eds., *Proceedings of Formal Grammar 06*. CSLI Publications.
- Baker, M. 1997. Thematic Roles and Syntactic Structure. In L. Haegeman, ed., *Elements of Grammar, Handbook of Generative Syntax*, pages 73–137. Dordrecht: Kluwer.

- Curien, P. and H. Herbelin. 2000. Duality of computation. In I. 2000, ed., *Proceedings of the Fifth AGM SIGPLAN*, pages 233–243. Montreal, Canada: SIGPLAN Notices.
- Davidson, D. 1966. The Logical Form of Action Sentences. In D. Davidson, ed., *Essays on Actions and Events*. Oxford: Clarendon Press.
- de Groote, P. 1996. Partially commutative linear logic: sequent calculus and phase semantics. In M. Abrusci and C. Casadio, eds., *Proofs and Linguistic Categories*, pages 199–208. CLUEB - University of Chieti.
- de Groote, P. 2001a. Towards Abstract Categorical Grammars. In ACL'01 (2001), pages 148–155.
- de Groote, P. 2001b. Type raising, continuations, and classical logic. In *Proceedings of the 13th Amsterdam Colloquium*, pages 97–101. Amsterdam: Language and Computation.
- de Groote, P. 2007. Towards a Montagovian Account of Dynamics. In *Proceedings of Semantics and Linguistic Theory XVI*. CLC Publications.
- Hale and Keyser. 1993. On Argument Structure and the Lexical Expression of Syntactic Relations. In Hale and Keyser, eds., *The View from Building 20*. Ithaca: MIT Press.
- Herbelin, H. 2005. *Au coeur de la dualité*. Phd thesis, Université Paris 11.
- Kratzer, A. 1994. External arguments. In E. Benedicto and J. Runner, eds., *Functional Projections*. Amherst: University of Massachusetts, Occasional Papers.
- Lambek, J. 1958. The Mathematics of Sentence Structure. *American Mathematical Monthly* 65:154–170.
- Lecomte, A. 2005. Categorical grammar for minimalism. In P. S. C. Casadio and R. Seely, eds., *Language and Grammar, Studies in Mathematical Linguistics and Natural Language*, pages 163–188. Stanford: CSLI Publications.
- Lecomte, A. and C. Retoré. 2001. Extending Lambek grammars: a logical account of Minimalist Grammars. In ACL'01 (2001), pages 354–362.
- Moortgat, M. 1997. Categorical type logics. In van Benthem and ter Meulen (1997), chap. 2, pages 93–178.
- Morrill, G. 1994. *Type Logical Grammar, Categorical Logic of Signs*. Dordrecht: Kluwer.
- Muskens, R. 1996. Combining Montague Semantics and Discourse Representation. *Linguistics and Philosophy* 19:143–186.

- Muskens, R. 2003. Languages, lambdas and logic. In G.-J. Kruijff and R. Oehrle, eds., *Resource Sensitivity in Binding and Anaphora*. Amsterdam: Kluwer.
- Parigot, M. 1992. $\lambda\mu$ -calculus: an algorithmic interpretation of classical natural deduction. In A. Voronkov, ed., *Proceedings of the International Conference on Logic Programming and Automated Reasoning*. Berlin: Springer.
- Pollard, Carl. 2007. Convergent Grammars. Tech. rep., The Ohio State University.
- Pollard, Carl. 2008. Covert movement in logical grammar. *ESSLLI Workshop on Ludics, Symmetric calculi and Continuations*.
- R. Muskens, J. van Benthem and A. Visser. 1997. Dynamics. In van Benthem and ter Meulen (1997), chap. 10, pages 587–648.
- Ranta, A. 1994. *Type Theoretical Grammar*. Oxford University Press.
- Stabler, E. 1997. Derivational minimalism. In C. Retoré, ed., *Logical Aspects of Computational Linguistics*, vol. 1328 of *LNCS/LNAI*, pages 68–95. Springer.
- Stabler, E. 2001. Recognizing head movement. In P. de Groote and G. Morrill, eds., *Logical Aspects of Computational Linguistics*, vol. 2099 of *LNCS/LNAI*, pages 245–260. Springer.
- van Benthem, J. and A. ter Meulen, eds. 1997. *Handbook of Logic and Language*. Elsevier.
- van Eijck, J. 2007. Context and the composition of meaning. In H. Bunt and R. Muskens, eds., *Computing Meaning 3*, pages 173–193. Netherlands: Springer.
- Vendler, Z. 1967. Verbs and Times. In Z. Vendler, ed., *Linguistics in Philosophy*. Ithaca: Cornell University Press.
- Vermeulen, C. F. M. 1993. Merging without Mystery or: Variables in Dynamic Semantics. *Journal of Philosophical Logic* 24(4):405–450.
- Zeevat, H. 1991. *Aspects of Discourse Representation Theory and Unification Grammar*. Phd thesis, Amsterdam University.

