A Linear-Logical approach to some Syntactico-Semantic phenomena in Romance Languages

Alain Lecomte
GRIL- Université de Clermont-Ferrand
(France)

Abstract

Non-commutative linear logic is used in this paper in order to give a representation of some syntactico-semantic problems which occur mainly in romance languages like: discontinuous constituants (see the negation in French for instance) and cliticization. The central idea is that several processes can be achieved in parallel when parsing a sentence, for instancea process of consumption of valencies and a process of synchronization between separate parts of a discontinuous sign. When signs are properly designed, ungrammatical sentences like: * *je vois ne pas Marie* or * *je vois Marie ne pas* are ruled out, and in the same veine, sentences like **il lui le donne* or * *il le donne lui* . Moreover, using linear-logical operators allows us to obtain a nice representation of the process of production of a semantic interpretation. The whole enterprise belongs to the "grammar as proof-theory"-paradigm.

En el presente articulo se utiliza la logica lineal no conmutativa para dar una representacion de algunos problemas sintactico-semanticos que ocurren preferentemente en las lenguas romànicas, tales como constituyentes discontinuos (por ejemplo, el caso de la negacion en francés) o los cliticos. La idea central es que se pueden realizar varios procesos en paralelo cuando se hace un parsing de una oracion y que la logica lineal ha sido creada justamente para describir procesos paralelos. Asi por ejemplo un proceso de consumicion de valencias y un proceso de sincronizacion entre partes separadas de un signo discontinuo se pueden realizar en paralelo. Si los signos son diseñados correctamente, quedan excluidas oraciones agramaticales como: * *je vois ne pas Marie* , o * *je vois Marie pas* ; e igualmente quedan excluidas oraciones como: * *il lui le donne* o * *il le donne lui* , etc. Ademas, el uso de operadores logico-lineales, nos permite representar el proceso de produccion de una interpretacion semantica de una manera elegante. El presente trabajo se incluye en el paradigma "grammar as proof-theory".

1. Introduction

1.1. The logical approach

A very promising approach to grammar consists in embedding a notion of grammar into a much more general framework. Why a logical framework? There are many misunderstandings on this point. Many linguists think of it as a kind of reductionism and ask: what is logical in essence in syntax, for instance? In saying that, they miss an important point, in our opinion. Logic is no longer the same as it was in the ancient days. In particular, new logics like linear logic (J.Y. Girard [5]) are not very concerned by truth values! They are not even very concerned by set-theoretical interpretations. Their kind of semantics is not a tarskian one but a Heyting semantics. Briefly, they are mere systems of description of the way information is produced, communicated and consumed. It is the reason why we feel authorized to use the systems they provide as tools for describing natural languages, which are also, after all, information systems. Moreover, we aim at using general methods, coming from this more general framework, for solving specific problems of Natural Language Processing.

1.2. Resource-sensitive logics

Among logical systems which belong to the family of resource-sensitive logics, the Lambek calculus has been intensively studied in the past (Lambek [9] , Lambek [10] , Moortgat [11]), but pure Lambek calculus does not provide any account for linguistic phenomena which occur frequently in ordinary language, like relatively free word order, discontinuity or gapping phenomena (see Moortgat [12], Moortgat& Morrill [14], Morrill [15]). Recently, many new devices have been added to this pure calculus in order to make it more sensitive to such kinds of phenomena, like permutability, limitations to associativity and so called "structural modalities" (Morrill [15], Hepple [7]). Another trend of research has consisted in making "semantics intervene into syntax", according to the expression used by Gabbay. This way of doing is much inspired by Labelled Deductive Systems (Gabbay [4]) and consists in putting into signs indications refering to the algebraic structure which is used as the semantics of the formulae. We are trying here to give another alternative for dealing with these phenomena. Moreover, this perspective is able to provide representations for other phenomena, which are not in general dealt with in the frame of categorial grammar and Lambek calculus. For instance, we aim at using it to give a solution to the problem of adverbs which behave syntactically as adjuncts and semantically as heads, (cf Dalrymple & al. [3]). In fact, this perspective amounts to go beyond the scope of Lambek calculus and to include more material coming from *non-commutative linear logic*. We elaborate a new linguistic model where words and expressions behave like small processes in a complex architecture.

2. Linear-logical preliminaries

2.1. The multiplicatives

Let us begin by the multiplicative part of linear logic. It is now well known that linear logic is obtained when removing structural rules from the classical sequent calculus. Commutative linear logic only deletes the weakening rule and the contraction rule. Non-commutative linear logic removes also the permutation rule. Thus non-commutative linear logic meets the Lambek calculus, but in the latter, the only connectives are /, \ and • (product). The connectives / and \ may be written as oriented implicational operators: o-- and --o . We have here to recall the main feature which distinguishes linear implication from the classical one. When a premise A is used once jointly with an implicative A --o B , A is removed from the set of premises: we can say that A is consumed. Thus, for instance, it is not possible to derive B from {A, A --o (A --o B)} . Moreover, all the premises must be consumed in a deduction. For instance the formula A --o (B --o A) is not a theorem.

The product corresponds to the non commutative product ("•" in Asperti [2] , "<", read: "precede" in Retoré [18]). Non commutative multiplicative linear logic adds the connectives ℘ ("par "), ⊥ (duality)[1] and the exponentials: ! ("of course") and ? ("why not?"). And the whole system contains also the additives: & and ⊕.

2.2. The exponentials

Rules for exponentiation :

$$\frac{\Gamma \vdash \Delta}{\Gamma, !A \vdash \Delta}\ W! \qquad \frac{\Gamma, !A, !A \vdash \Delta}{\Gamma, !A \vdash \Delta}\ C! \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma, !A \vdash \Delta}\ L! \qquad \frac{!\Gamma \vdash A, ?\Delta}{!\Gamma \vdash !A, ?\Delta}\ R!$$

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash ?A, \Delta}\ W? \qquad \frac{\Gamma \vdash ?A, ?A, \Delta}{\Gamma \vdash ?A, \Delta}\ C? \qquad \frac{!\Gamma, A \vdash ?\Delta}{!\Gamma, ?A \vdash ?\Delta}\ L? \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash ?A, \Delta}\ R?$$

Operators are interdefinable. We have for instance, in commutative logic:

$$A \multimap B \equiv A^{\perp} \wp B$$

$$(?A)^{\perp} \equiv !(A^{\perp})$$

$$(!A)^{\perp} \equiv ?(A^{\perp})$$

In non-commutative logic, we have to distinguish between ${}^{\perp}()$ and $()^{\perp}$. We have for instance (Abrusci [1]):

$$A \circ\!\!- B \equiv A \wp B^{\perp}$$

2.3. The products

We use "⊗" as a notation for the commutative product[2]. Because of the removal of the permutation rule, commas of the left-hand side of a sequent have the interpretation of a • (non-commutative product). As usual, commas on the right-hand side are interpreted as ℘ (which is assumed here non commutative). ⊗ will be used inside a sign in order to agglomerate its components. It makes sense to assume that components of a same sign are permutable.

2.4. A computational interpretation

According to the computational interpretation of Linear Logic (cf Asperti [2] and Retoré [18]) sequents are correct if and only if they correspond to correct communications between processes. Formulae are processes, and proof-nets (a proof-net is a geometrical device which expresses the essence of a proof) are interpreted as communication structures between them, along channels represented by links. According to Asperti's interpretation : "the main notion is that of system. A system is a distributed process which is not supposed to have any communication with the surrounding world. A system is composed by

---

[1] Let us pay attention to the fact that, in non-commutative linear logic, there are two instances of ⊥ : ${}^{\perp}A$ and $A^{\perp}$.

[2] we can define A⊗B by: A⊗B ≡ (A•B)&(B•A), that means: a choice to be made between (A•B) and (B•A)

processes, connected together by links. Processes look as terms of a process algebra." This kind of system is made of the following operators and atoms:

- atoms : A, B, ..., and their duals $A^{\perp}$, $B^{\perp}$, ...
- parallel composition : A ℘ B
- sequential composition : A • B
- internal non-determinism : A & B
- external non-determinism : A ⊕ B
- mutual exclusion : A ⊗ B

Synchronization is the only possible move at the level of atoms, it occurs between an atom and its dual. Links are put between processes according to the kind of operators, and conventions are given for the circulation of tokens along these links. The true nature of parallelism and sequentialisation is given through these conventions. A sequent thus expresses a way in which processes are combined. A good combination is a deadlock free structure. Methods for checking the deadlock-freeness meet the proof-net method in linear logic.

In the light of these computational interpretations, we can interpret neutral elements. As we know, multiplicative connectives have their neutral elements: **1** for ⊗ and ⊥ for ℘ . According to the interpretation in terms of resources, we can say that **1** corresponds to "no resource", and ⊥ to a process that consumes nothing: as soon it is activated, it terminates. It is the reason why a sequent with empty succedent has an interest in our frame: it says that all the processes on the left-hand side work altogether as an isolated system, they have no output, and they work as if they were the particular process. Finally, we must notice that it is possible to choose an interpretation of linear logic where **1**=⊥. This corresponds to the case where the process which does not consume anything is at the same time the process which does not produce anything[3].

2.5. Signs

Complex types will be assigned to words and morphemes. We will sometimes call them signs according to the trend initiated in the modern days by Pollard & Sag ([17]) and in the ancient days by Saussure. These signs include two parts: a phonological (or prosodic) one, realized as a string, and a proper type which is a linear-logical formula.

Examples:

- *Pierre* ------> `pierre` :np
- *aime* -------> `aime` : (np\s)/np
- *il* --------> `il` : (s/(np\s)) ℘ $(?np)^{\perp}$ (see below)

Morrill ([15]) and Moortgat ([13]) give a complete formulation of a sign-based labelled deductive system. We allege this paper by not mentioning the strings.

---

3 Let us notice that the identity **1** = ⊥ is equivalent to the fact that the empty sequent is a theorem. In this context, we obtain for each A the following sequents as theorems: |–?A , B/?A A |– B and B/?A |– B. The exponential "?" truely indicates optionality. Deductions: we get |–?A by W?. We get B/?A A |–B from A |–?A and B |– B , and B/?A |– B from |–?A and B |– B.

2.6. Generalized connectives

Linear Logic rules are not recalled here. Let us only notice the ability to generalize the multiplicative connectives according to the definition by Troelstra ([19]):

> A generalized multiplicative connective C is associated to a set of partitions, and must have a dual connective, with the same number of arguments, allowing us to extend the De Morgan duality by :
>
> $C(A_1, ..., A_n)^{\perp} \equiv C^*(A_1{}^{\perp}, ..., A_n{}^{\perp})$

We will then introduce "**`par`**" for types of different dimensions (denoted here by different fonts of characters, or we could say "colours"). These **`par`**-connectives will have the property of splitting sequents into subsequents by picking up parts of signs which belong to a same dimension and making a one-dimensional sequent to demonstrate (when seeing deductions from the bottom). In this splitting, strings attached to types are inherited from types to subtypes. When reading a deduction from the top, subsequents are merged at each **`par`**-step (or **`mix`**-step). But places of formulae are not undetermined because of this inheritance convention. For instance a sign `il` : s/(np\s) and a sign il : $(?np)^{\perp}$ can only merge into the sign: `il` : s/(np\ s) $\wp$ $(?np)^{\perp}$ because this latter sign makes sense according to the lexicon and because a sign `il` :$(?np)^{\perp}$ cannot appear alone in a terminal sequent. We shall express this mechanism by the following kind of rule:

$$\frac{A_{i1}, ..., A_{im}, x : A, B_{i1}, ..., B_{in} \vdash s : C \qquad \mathbf{A_{j1}, ..., A_{jk}}, x : \mathbf{B}, \mathbf{B_{j1}, ..., B_{jl}} \vdash}{A_1, A_2, ..., A_p, x : A \wp_i \mathbf{B}, B_1, ..., B_q \vdash s : C} L\wp_i$$

(where $\{\{A_{i1}, ..., A_{im}\}, \{A_{j1}, ..., A_{jk}\}\}$ is a partition of $\{A_1, A_2, ..., A_p\}$ and $\{\{B_{i1}, ..., B_{in}\}, \{B_{j1}, ..., B_{jl}\}$ a partition of $\{B_1, ..., B_q\}$ and where the second premise is only made of unidimensional signs having the same dimension).

3. An example of discontinuous constituant: the French negation

3.1. Introduction

Usual approaches of discontinuity in computational linguistics (for instance in unification grammars) are rather procedural ones (the sign made by incorporating a negation particle into the verbal node is made waiting for the other particle). It seems that such an approach may be avoided provided that we are able to conceive parallel deductions. The process of negation can thus be described by assuming at least two dimensions of the signs. One dimension concerns concatenation phenomena and another one concerns the consumption of valencies. It is well known that the verbal valencies are not modified when adjoining a negation to the verb. This can be expressed by assigning a modifier-type to the negation particle

ne: `ne` ----> (np\s)/(np\s)

but only in the second dimension of the sign. The other characteristic feature of *ne* is its waiting for another negation particle (*pas, plus, rien, jamais*), it is a concatenation property of *ne* and it is concerned with the first dimension of the sign. Very roughly, the sentence is correct if and only if two things are simultaneously checked: first, a correct reduction to the primitive type **`s`**, second, the good "synchronization" between negation particles. These two mechanisms must be processed in parallel. The idea is therefore to assign to *ne* a complex type which includes two subtypes, connected by a **par** -connective: *ne* ------> [(np\s)/(np\s)] $\wp$ **neg** and to *pas* the single type: *pas* ----> $^{\perp}$**neg**.

3.2. Some facts

Let us suppose we want to describe the following facts. Sentences or phrases like (1) to (4) are correct in French but sentences (5) to (9) are not.

1. Pierre voit Marie

2. Pierre ne voit pas Marie

3. Pierre n 'a pas vu Marie

4. ne pas voir Marie

*5. *Pierre pas ne voit Marie*

*6. *Pierre ne pas voit Marie*

*7. *Pierre ne voit Marie pas*

*8. *Pierre n 'a vu pas Marie*

*9. * ne voir pas Marie*

Laws can be formulated like: (1) *ne* must stand just before the verb in any case, (2) *pas* must be placed after finite tensed verbs and before infinitives, (3) *pas* must be agglomerated to the verb before its non-subject complements are consumed.

3.3. Adverbials and negation

(1) is taken into account thanks to the "consumption of valencies" dimension (like in ordinary categorial grammar). But let us now try to adress the case of (2). For that, we may suppose that french tensed verbs optionally admit a quantificational adverb on their right, that pas behaves like such an adverb and that this adverb must be adjacent to the verb. This assumption can be expressed by assuming an atom advq (for quantificational adverb) which is asked for by the verb. This amounts to assign a type np\s/np/advq to a transitive verb like *aimer* or *voir*. Moreover, because this kind of adverb is optional, we assign the type np\s/np/?advq to such a tensed transitive verb. And we add the following assignments for the negative particles:

ne ------> [(np\s)/(np\s)] $\wp$ **neg**

pas ------> advq $\otimes$ $^{\perp}$**neg**

The following deduction so obtains for *Pierre ne voit pas Marie*.

```
                              np |– np              s |– s
                              ------------------------------\L
                                       np np\ s |– s      np |–np             s |– s
                                       ---------------- \R   -------------------------------\L
                                       np\s |– np\s              np np\s |– s
                                       ---------------------------------------------- /L
advq |– advq                                     np (np\s)/(np\s) np\s |– s           np |– np
------------------ ?                             ------------------------------------------------------- /L
 advq |–?advq                                             np (np\s)/(np\s) np\s/np np |– s
------------------------------------------------------------------------------------------------------------- /L
          np (np\s)/(np\s) np\s/np/?advq advq np |– s                              neg ⊥neg |–
          ------------------------------------------------------------------------------------------------------------------------ ℘L
                              np ((np\s)/(np\s)) ℘ neg    np\s/np/?advq  advq⊗⊥neg  np |– s
```

It is easy to check that *Pierre ne voit pas beaucoup Marie* is also accepted but not sentences like:

* *Pierre ne voit Marie pas*

* *Pierre ne voit pas Marie beaucoup*

* *Pierre ne pas voit Marie*

3.4. The infinitive and past-participle cases

Now, in order to cover the infinitive case, we may assume that infinitive forms have types ?advq\np\s/np and that adding a tense has as a consequence to displace the missing ?advq from left to right. Concerning (3) and (8), we introduces two new features: the type assigned to an auxiliary like *avoir* , and the type assigned to the past participle. Hence, the following assignment:

vu ------> (np\s/np) ⊗ **pp**

a -------> ((np\s)/(np\s)) ⊗ **pp**$^{\perp}$ ℘ (?advq)$^{\perp}$

4. Remark on Semantic Representations

4.1. Semantics with syntax

Inside a sign with type A ℘ B or A ⊗ B, the communication between A and B can be expressed by means of a shared variable X. Individual variables will be use for building semantic representations. Let us assume that each kind of phrase admits a head. We interpret a head as bearing a **`par`** between a syntactic part and a semantic one: it triggers a parallelisation of treatments. For instance, assuming the verb as the head of a sentence, the verb is a parallelisation of a syntactic part and a semantic part. For instance, we will have the following type for a transitive verb like *lire* : ∀X. ∀Y.((np ℘ X)\s/(np ℘ Y)) ℘ lire(X, Y) . The semantic representation of the sentence will thus be obtained as something provided in parallel with the syntactic type in the succedent of the resulting sequent. Let us take as an example the sentence *Pierre lit Nana* (Pierre reads Nana), we can perform the following deduction:

```
                                              np |– np              nana* |– nana*
                                              ----------------------------------------
np |– np          pierre* |– pierre*          np ℘ nana* |– np ℘ nana*          s |– s
----------------------------------------      ----------------------------------------------
np ℘ pierre* |– np ℘ pierre*                    s / (np ℘ nana*), np ℘ nana* |– s
-----------------------------------------------------------------------------------------
      np ℘ pierre*, (np ℘ pierre*)\s/(np ℘ nana*), np ℘ nana* |– s        lire(p*, n*) |– lire(p*, n*)
      ----------------------------------------------------------------------------------------------
      np ℘ pierre*, ((np ℘ pierre*)\s/(np ℘ nana*)) ℘ lire(pierre*, nana*), np ℘ nana* |–s ℘ lire(p*, n*)
      ----------------------------------------------------------------------------------------------
      np ℘ pierre*, ∀Y.((np ℘ pierre*)\s/(np ℘ Y)) ℘ lire(pierre*, Y), np ℘ nana* |– s ℘ lire(pierre*, nana*)
      ----------------------------------------------------------------------------------------------
      np ℘ pierre*,  ∀X. ∀Y.((np ℘ X)\s/(np ℘ Y)) ℘ lire(X, Y), np ℘ nana* |– s ℘ lire(pierre*, nana*)
```

which instantiates the variables inside the semantic formula assigned to the verb.

4.2. The need for second-order rules

This process of instanciation can be represented by introducing second-order quantifier rules, like:

$$\frac{\Gamma,\ A[t],\ \Gamma' \vdash \Delta}{\Gamma,\ \forall X.\ A[X/t],\ \Gamma' \vdash \Delta}\ \forall L$$

$$\frac{\Gamma \vdash \Delta,\ A[t],\ \Delta'}{\Gamma \vdash \Delta,\ \exists X.\ A[X/t],\ \Delta'}\ \exists R$$

5. Clitics

5.1. Clitics in Categorial Grammar

Most implementations of categorial grammars (like for instance Klein, Zeevat & Calder ([8])) consider clitics as type-raised signs and use polymorphism. For instance, we could assign to the french clitic *le* the categorial type:

X/(X/np[obj])

(where X is a type-variable and np[obj] a parametrized type (a np with feature `obj`)) in order that the sequence:

le mange

in *Pierre le mange* (Pierre eats it) reduce to a new verbal sign: np[nom]\s according to the application rule and the instanciation principle:

X/(X/np[obj]) (np[nom]\s)/np[obj] -----> np[nom]\s

(because X is instanciated to np[nom]\s). Other treatments use type-raised signs and structural modalities, like Morrill & Gavarro ([16]) who treat the problem of catalan clitics. These implementations are correct on a point: clitics are analyzed as signs which give new verbal signs from other ones. But they miss one important point: when using a clitic-object like *le* , the object-valency of the verb is not necessarily consumed, because it can re-appear as a lexical NP which is, in this case, *topicalized* like in:

Pierre le mange, ce gateau

le gateau, Pierre le mange

It is therefore important to try another solution which could take so topicalized NPs into account.

5.2. The informational dimension

Many linguists in the past (like Halliday [6]) have insisted on the fact that several systems are superposed in language. For instance, Halliday says: "Theme is concerned with the information structure of the clause, with the status of the elements [...] as components of a message". Thus, beside a purely syntactic dimension, we have to assume a kind of informational system which is only concerned by the hierarchisation of information in the sentence. For instance, when I say : *mon frère, son vélo, il y a les freins qui ne marchent plus* (my brother, his bike, the brakes no longer work), I implicitely give information on the order of the topics: first I speak of my brother, second of his bicycle and third I give an information on the brakes. This dimension is relatively free with respect to the purely syntactic one. We propose as a matter of hypothesis that types associated to referential nominals are disjunctive. It is possible to use them in a purely "informational" manner, and in this case, they must be replaced by clitics in the pure syntactic dimension, or they can be syntactically used and clitics are then useless. When they are informationnally used, clitics are imposed in order to fulfill the requirements involved by the process of consuming valencies. This perspective involves the &-rules, where & is the additive connective of Girard, which can be interpreted as a choice to be made (or *internal choice*).

• Rules for the &-connective:

$$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\;\&R \qquad \dfrac{\Gamma, A \vdash \Delta}{\Gamma, A \,\&\, B \vdash D}\;\&L1 \qquad \dfrac{\Gamma, B \vdash \Delta}{\Gamma, A \,\&\, B \vdash \Delta}\;\&L2$$

Let us give the following assignment to a noun and a clitic:

Pierre -------> np&np

il -------> $[\mathrm{s}/(\mathrm{np}\backslash \mathrm{s})] \wp\ \mathit{np}^{\perp}$

and we obtain the following deduction for the sentence:

Pierre, il regarde la mer : (Pierre, he looks at the sea)

$$\dfrac{\dfrac{\mathrm{s}/(\mathrm{np}\backslash\mathrm{s}),\ \mathrm{np}\backslash\mathrm{s}/\mathrm{np},\ \mathrm{np} \vdash \mathrm{s} \qquad \mathit{np},\ \mathit{np}^{\perp} \vdash}{\mathit{np},\ [\mathrm{s}/(\mathrm{np}\backslash\mathrm{s})] \wp\ \mathit{np}^{\perp},\ \mathrm{np}\backslash\mathrm{s}/\mathrm{np},\ \mathrm{np} \vdash \mathrm{s}}\;\wp L}{\mathrm{np}\&\mathit{np},\ [\mathrm{s}/(\mathrm{np}\backslash\mathrm{s})] \wp\ \mathit{np}^{\perp},\ \mathrm{np}\backslash\mathrm{s}/\mathrm{np},\ \mathrm{np} \vdash \mathrm{s}}\;\&L$$

By assuming that a nominal phrase is a np&*np* we assume that it can be either a np (consumable in the system of valencies) or a *np* (which can be synchronised with another element in the subsystem of hierarchisation of information). The clitic contains in parallel two informations: one concerning the process of consuming valencies and the other concerning a synchronisation with a possible antecedent. Moreover, we can stipulate that deductions which belong to particular dimensions can use or cannot use such and such structural rules. For instance, it is convenient to assume that the permutation rule is applicable in the informational dimension. In this case, as we know, $A^{\perp}$ and ${}^{\perp}A$ do collapse into only one negation, $A^{\perp}$.

5.3. Further examples

Let us go further on this topic and let us show that when giving the following assignment:

• subject pronoun:

il ------> $s/(np[nom]\backslash s) \wp\ ?np[nom]^{\perp}$

• object pronoun:

le ------> $(np[nom]\backslash s)/((np[nom]\backslash s)/np[obj]) \wp\ ?np[obj]^{\perp}$

• dative pronoun:

lui ------> $((np[nom]\backslash s)/np[obj])/((np[nom]\backslash s)/np[dat])/?np[obj] \wp\ ?np[dat]^{\perp}$

the following sentences are correct:

(1) *il le lui donne*

(2) *il le donne*

(3) *il le lui donne, à Marie*

(4) *il le donne, son vélo*

(5) *il le lui donne, son vélo, à Marie*

and that the following ones are not:

(6) * *il lui donne*

(7) * *il lui le donne*

Example: (4)

```
                                                               np[n] |– np[n]          s |– s
                                                               --------------------------------
                                      np[o] |– np[o]    [ ] |– ?np[d]     np[n],np[n]\s |– s
                                      --------------    -----------------------------------
                                      np[o] |– ?np[o]       np[n],(np[n]\s)/?np[d] |– s
                                      -----------------------------------------------
                                        np[n],(np[n]\s)/?np[d]/?np[o], np[o] |– s                  …
                                      -----------------------------------------------       --------------
np[o] |– np[o]                        (np[n]\s)/?np[d]/?np[o], np[o] |– np[n]\s               np[n]\s |– np[n]\s
--------------                        -----------------------------------------        ------------------------
np[o] |– ?np[o]   <emp> |– <emp>    (np[n]\s)/?np[d]/?np[o] |– (np[n]\s)/np[o]   s/(np[n]\s), np[n]\s |– s
---------------   --------------    ----------------------------------------------------------------------
 np[X] |–?np[o]  <emp>|–?np[n]      s/(np[n]\s), (np[n]\s)/(np[n]\s)/np[o], (np[n]\s)/?np[d]/?np[o] |– s
---------------  ---------------------------------------------------------------------------------------
?np[o]⊥,np[X] |–    (s/(np[n]\s)) ℘ ?np[n] , (np[n]\s)/(np[n]\s)/np[o], (np[n]\s)/?np[d]/?np[o] |– s
--------------------------------------------------------------------------------------------------------
   (s/(np[n]\s)) ℘ ?np[n] , ((np[n]\s)/(np[n]\s)/np[o]) ℘ ?np[o]⊥, (np[n]\s)/?np[d]/?np[o], np[X] |– s
```

le cannot be added to the verb before *lui* like in (7) because when doing so, when *le* applies to the verb *donner* (to give), the optional valency ?np[dat] is removed, but this valency is required for any application of *lui* . (6) is also ruled out by the actual type assigned to *lui* : when the verbal form is consumed by the clitic *lui* , a new verbal form is produced, for which the object-valency becomes obligatory. Hence, this latter will have to be consumed by something (either a clitic-object or a NP object).

6. Placement of adverbs

Our last example will concern the placement of adverbs, according to the problems exposed in Dalrymple & al. [3]. We resume the questions by saying that in general adverbs are syntactically treated like adjoints even if they often play the role of a semantic head. For instance, in *Pierre aime évidemment Marie* (Pierre obviously loves Marie), the adverb *évidemment* semantically becomes a predicate with sentential argument, while syntactically it is inserted inside the verbal phrase. We propose the following assignment:

évidemment ------> $\forall Z.\, {}^{\perp}Z \wp$ évidemment(Z)

Deduction:

$$\dfrac{np \vdash np \quad p^* \vdash p^*}{np \wp p^* \vdash np \wp p^*} \qquad \dfrac{\dfrac{np \vdash np \quad m^* \vdash m^*}{np \wp m^* \vdash np \wp m^*}}{s/(np \wp m^*),\ np \wp m^* \vdash s} \qquad \dfrac{\dfrac{a(p^*, m^*) \vdash a(p^*, m^*)}{a(p^*, m^*),\ {}^{\perp}a(p^*, m^*) \vdash} \quad e(a(p^*, m^*)) \vdash e(a(p^*, m^*))}{}$$

$$\dfrac{np \wp p^*,\ np \wp p^* \backslash s/(np \wp m^*),\ np \wp m^* \vdash s \qquad a(p^*, m^*),\ {}^{\perp}a(p^*, m^*) \wp e(a(p^*, m^*)) \vdash e(a(p^*, m^*))}{np \wp p^*,\ (np \wp p^*) \backslash s/(np \wp m^*)) \wp a(p^*, m^*),\ {}^{\perp}a(p^*, m^*) \wp e(a(p^*, m^*)),\ np \wp m^* \vdash s \wp e(a(p^*, m^*))}$$

$$\dfrac{\dfrac{np \wp p^*,\ (np \wp p^*) \backslash s/(np \wp m^*)) \wp a(p^*, m^*),\ \forall Z. {}^{\perp}Z \wp e(Z),\ np \wp m^* \vdash s \wp e(a(p^*, m^*))}{np \wp p^*,\ \forall Y.(np \wp p^*) \backslash s/(np \wp Y)) \wp a(p^*, Y),\ \forall Z. {}^{\perp}Z \wp e(Z),\ np \wp m^* \vdash s \wp e(a(p^*, m^*))}}{np \wp p^*,\ \forall X.\ \forall Y.(np \wp X) \backslash s/(np \wp Y)) \wp a(X, Y),\ \forall Z.\ {}^{\perp}Z \wp e(Z),\ np \wp m^* \vdash s \wp e(a(p^*, m^*))}$$

7. Conclusion

According to this new approach, which considerably extends traditional categorial grammars, we achieve the goal of treating linguistic parsing as a deduction process. Moreover, we showed that, by distinguishing several dimensions of a sign $A_1, ..., A_n$, connected together by special connectives (the paR-connectives), we are able to desintricate many processes which are generally working altogether in the production/recognition of a sentence. Our conception of a sign like an object: σ: $A_1 \wp ... \wp A_n$ allows to derive deductions by means of inference rules which desintricate dimensions in order that subdeductions are derived inside them according to three main kinds of mechanisms: consumption, synchronization and instanciation. In this view, we can say that the process of parsing a sentence amounts to consume information associated with words in order to produce a meaning, a view which seems highly desirable.

References

[1] M. Abrusci. Exchange connectives for non commutative classical linear propositional logic. Preprint. Universita di Roma La Sapienza, 1993.

[2] A. Asperti. A Linguistic approach to Deadlock. Rapport de recherches LIENS-91-15, LIENS, 1991, Paris.

[3] M. Dalrymple, J. Lamping V. Saraswat. LFG Semantics via Constraints. *Proceedings of EACL*, Utrecht, 1993.

[4] D. Gabbay. *Labelled Deductive Systems* . Draft. Oxford University Press, 1991 (to appear).

[5] J. Y. Girard. Linear logic. *Theoretical Computer Science* , **50** , 1987, 1-102.

[6] M.A.K. Halliday. Notes on transitivity and theme in English.1967.

[7] M. Hepple. *The Grammar and Processing of Order and Dependency, a categorial approach*. PhD Thesis. Centre of Cognitive Sciences, Edinburgh, 1990.

[8] E. Klein, H. Zeevat, J. Calder. Unification Categorial Grammar. in *Categorial Grammar, Unification Grammar,and Parsing* N.J. Haddock, E. Klein, G. Morrill (eds). Edinburgh, Centre for Cognitive Science, 1987, pp. 195-222.

[9] J. Lambek. The Mathematics of Sentence Structure. *American Mathematical Monthly*, **65**, pp 154-170, 1958.

[10] J. Lambek. On the Calculus of Syntactic Types, *American Mathematical Society*, 1961.

[11] M. Moortgat. *Categorial Investigations*. Foris Pub. 1988.

[12] M. Moortgat. Generalized Quantifiers and discontinuous type constructors. OTS working papers. OTS-WP-CL-92-001. Utrecht, 1992. To appear in *Discontinuous Constitency*, W. Sijtsma, A. van Horch (eds). Mouton de Gruyer. Berlin.

[13] M. Moortgat. Labelled Deductive Systems for categorial theorem proving. OTS working papers. OTS-WP-CL-92-003. Utrecht, 1992. To appear in *Proceedings of the 8th Amsterdam Colloquium*, Dekker & Stokhof (eds).

[14] M. Moortgat & G. Morrill. Heads and Phrases. Type Calculus for Dependency and Constituent Structure. OTS working papers. RUU. Utrecht, 1991.

[15] G. Morrill. *Type Logical Grammar*. Report de Recerca, Departament de Llentguages i Sistemes Informatica, Universitat Politecnica de Catalunya, Barcelona, 1992.

[16] G. Morrill & A. Gavarro. Catalan Clitics. in *Word Order in Categorial Grammar* , A. Lecomte (ed), ADOSA, Clermont-Ferrand, 1992.

[17] C. Pollard & I. Sag. *Information Based Syntax and Semantics*. vol 1. CSLI Lecture Notes, Stanford, 1987.

[18]□C. Retoré. *Réseaux et Séquents Ordonnés* . Thèse de Doctorat. Université Paris 7. 1993

[19] A. S. Troelstra. *Lectures on Linear Logic*. Lecture Notes. CSLI, Stanford, 1991.