La non-commutativité comme argument linguistique : modéliser la notion de phase dans un cadre logique

Maxime Amblard

La non-commutativité comme argument linguistique : modéliser la notion de phase dans un cadre logique

Maxime Amblard*

* *LORIA, UMR 7503, Université de Lorraine, CNRS, Inria Nancy Grand-Est Campus scientifique F54506 Vandoeuvre-lès-Nancy Cedex*

RÉSUMÉ. *L'une des questions du traitement automatique des langues est de discuter de la réalité de la capacité langagière des formalismes. Au delà de la modélisation linguistique, la théorie générative de Chomsky et le minimalisme s'intéressent à appréhender le langage humain en tant que processus cognitif, ce qui conduit à introduire le principe de dérivation par phases. Une première formalisation du minimalisme a été proposée dans (Stabler, 1997) afin, notamment, d'en étudier les propriétés computationnelles. L'extension formelle proposée ici, basée sur les Grammaires Minimalistes Catégorielles, (Amblard, 2011), s'attache à intégrer la notion de phase dans un cadre logique qui permet aussi de définir un calcul sémantique. Les enjeux de cette modélisation nous amènent à discuter de la commutativité et de la non-commutativité dans le formalisme.*

ABSTRACT. *One of the recurring questions in natural language processing is the models's ability to account for the reality of language ability. Chomsky's Generative Theory and Minimalism are interested in understanding human language as a cognitive process, which is especially highlighted in the latest proposals by the principle of derivation by phases. A first formalization of Minimalism was introduced in (Stabler, 1997) to study the computational properties. The extension proposed here attempts to account for the idea of phase in a logical framework that allows to easily define a semantic calculus from parsing. This approach raises the problem of using the commutativity and non-commutativity in the Minimalist Categorial Grammars, (Amblard, 2011).*

MOTS-CLÉS : *modélisation linguistique, logique, théorie générative, syntaxe, grammaires catégorielles, grammaires minimalistes, grammaires minimalistes catégorielles*

KEYWORDS: *linguistic modeling, logic, generative theory, syntax, Categorial Grammars, Minimalist Grammars, Minimalist Categorial Grammars*

3e soumission à *TAL 56-01*, le 07/06/2015

1. Introduction et motivation

L'analyse linguistique est naturellement confrontée à un double problème. D'un côté elle doit reconnaître une forme linguistique (par exemple une phrase) à partir d'une suite de mots, où l'ordre est porteur d'une information structurelle, et d'un autre côté mettre en relation ces mots au delà de leur proximité immédiate (pour rendre compte des relations à longue distance, ou encore de la modification d'ordre canonique classique). Le premier problème plaide en faveur de l'utilisation de la non-commutativité (où l'on respecte strictement les relations de proximité) et le second en faveur de la commutativité (par exemple les relations à longue distance).

Pour poser le vocabulaire, nous définissons la commutativité comme la propriété d'une opération qui permet de changer l'ordre des termes sans changer le résultat, par exemple $AB = BA$. Par opposition, la non-commutativité implique que $AB \neq BA$.

La commutativité et la non-commutativité sont régulièrement utilisées pour définir des cadres formels. Par exemple la non-commutativité est mobilisée dans les grammaires catégorielles (Ajdukiewicz, 1935) et (Bar-Hillel, 1953), HPSG (Sag et Pollard, 1987), les grammaires de pré-groupes (Lambek, 1999) ou encore LFG (Kaplan et Bresnan, 1982 ; Bresnan, 2001). Une autre approche est de définir des méta-structures simulant la non-commutativité comme dans les TAG (Joshi et Schabes, 1997) ou les grammaires d'interaction (Guillaume et Perrier, 2009). Mais là encore, les formalismes ne prennent pas simultanément en compte la commutativité et la non-commutativité. Il reste rare de discuter de leur utilisation et plus encore de leur coexistence dans des cadres formels pour la modélisation des langues naturelles.

Dans cet article, nous définissons un formalisme utilisant la commutativité et la non-commutativité en leur associant des usages distincts du point de vue linguistique. Nous utilisons un cadre logique pour modéliser la théorie générative de Chomsky (Chomsky, 1957). Dans un premier temps, nous revenons sur les définitions et les propriétés mobilisées qui motivent notre proposition.

1.1. *Éléments du minimalisme chomskien*

La théorie générative a connu de nombreuses évolutions depuis la proposition de synthèse qu'est le programme minimaliste (MP) (Chomsky, 1995). Dans le MP, Chomsky cherche à présenter une architecture permettant d'appréhender le langage humain comme un processus cognitif. La critique la plus fréquemment apportée est certainement le caractère non calculatoire de l'approche. Elle n'en reste pas moins riche d'une vaste littérature sur le versant linguistique. Le MP fait l'hypothèse de l'existence d'un calcul principal, porté par la syntaxe, qui permet de produire simultanément deux représentations, l'une correspondant à la suite de mots analysée (les chaînes de caractères), l'autre représentant la sémantique de l'énoncé.

Pour la théorie générative, l'analyse est lexicalisée, au sens où les informations utiles sont associées aux mots dans le lexique. Elle se base sur deux opérations : la fu-

sion (*merge*) et le déplacement (*move*). Afin de positionner les notions, sans présenter toute la théorie, nous considérons l'exemple très simplifié suivant :

(1) L'enfant lit un livre

L'opération de ***fusion*** permet de mettre en relation deux éléments pour construire un constituant. Par exemple le déterminant *un* est combiné avec le nom *livre* pour construire le constituant [un livre]. Pour motiver ce regroupement, la théorie utilise les traits associés aux items lexicaux correspondants. Ici, la position à droite ou à gauche (proximité et **non-commutativité**) permet la construction.

L'analyse est réalisée en plusieurs étapes, dont l'une correspond à la syntaxe profonde où le verbe reçoit ses arguments avant que ces derniers soient *déplacés* pour prendre leur position finale dans la réalisation de surface qui produit la suite de mots reconnue par l'analyse. Cette idée modifie profondément la notion de *parsing* en ce qu'il s'agit de chercher une dérivation dont la suite de mots reconnue seulement en fin de dérivation est la suite cherchée. Les productions intermédiaires ne sont pas évaluées. Un exemple où l'on observe un déplacement explicite est celui des questions (sans inversion du verbe et du sujet) :[1] :

(2) Quel livre l'enfant lit ?

La première partie de l'analyse de (2) est similaire à celle de (1) produisant :

[[l'enfant] lire [(un | quel) livre]]

La dernière partie de l'analyse utilise l'opération de déplacement pour faire passer l'objet devant le sujet et le verbe en supprimant un trait particulier de *quel*. Cette fois, le ***déplacement*** plaide pour l'utilisation de la **commutativité**.

[[Quel livre] [l'enfant] lire]

Les opérations du MP et de la théorie générative sont définies par les relations entretenues à l'intérieur d'un énoncé (ou d'un groupe de mots). Chaque syntagme est gouverné par un unique élément lexical (un mot), appelé la *tête* (*head*) qui induit la catégorie principale. Cette tête entretient soit une relation de *spécifieur*, soit de *complément* avec les autres éléments. La figure 1 reprend la structure générale d'un syntagme. Les *compléments* sont considérés comme des propriétés lexicales de la tête. Par exemple, le fait que *dire* accepte une complétive ne dépend pas de sa catégorie (verbe) mais bien de ses propriétés lexicales. A *contrario*, les *spécifeurs* sont des propriétés induites par la catégorie de la tête, par exemple le fait que le nom attende un déterminant pour construire un syntagme nominal.

La *projection maximale* d'une tête est définie comme le plus grand syntagme dont elle est la tête (soit le plus grand arbre d'analyse pour lequel elle reste la tête). Elle

1. Cet exemple est à nouveau une version simplifiée qui ne prend pas en compte l'inversion du verbe et du sujet, ce qui serait possible.

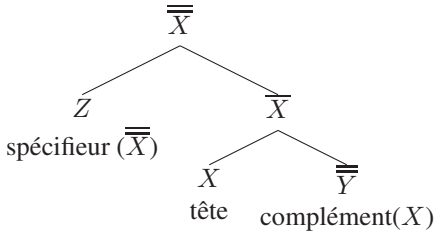

Figure 1. *Structure générale des syntagmes dans la théorie générative.*

admet trois niveaux d'analyse : [tête] (niveau 0), [tête [complément(s)]] (niveau 1) et [[spécifieur] tête [complément(s)]] (niveau 2).

Par ailleurs, dans le MP, Chomsky avance l'hypothèse que le processus conduisant à la représentation syntaxique se réalise sous forme de *phases* (Chomsky, 1999). En intégrant cette notion, Chomsky fait un pas de plus vers la modélisation cognitive. Il ajoute à l'analyse des moments permettant de valider ou d'invalider l'analyse en cours qui est réalisée au fur-et-à-mesure de la découverte de l'énoncé. Cette définition incrémentale de l'analyse est un avantage pour le TAL et ses applications. Techniquement, les *phases* définissent des contraintes (des îlots) : la *Phase Impenetrability Condition* (PIC) suppose que les éléments conservés à l'intérieur d'une phase ne sont plus accessibles pour la suite de l'analyse, a contrario des éléments déplacés en périphérie.

La dérivation débute à partir de l'item lexical du verbe. Elle fait évoluer ce noyau verbal à partir de plusieurs entrées lexicales tout au long de l'analyse. Ces entrées permettent de définir les *phases* et sont également utilisées dans l'interface syntaxe-sémantique pour rendre compte des UTAH[2]. Cette théorie est motivée par l'idée de rendre compte de processus cognitifs, notamment au travers de l'analyse linguistique, mais sans se placer dans un cadre formel explicite. Nous proposons une formalisation dans un cadre logique qui permet la manipulation simultanée de la commutativité et de la non-commutativité.

1.2. *De la logique comme outil de la formalisation*

La première formalisation du MP a été proposée dans (Stabler, 1997) avec les grammaires minimalistes (MG). Elles sont basées sur la notion de traits associés aux items lexicaux, pouvant se combiner pour déclencher les règles. À partir de cette proposition, les propriétés computationnelles du minimalisme ont été étudiées (Michaelis, 2009), et des algorithmes d'analyse efficaces proposés (Harkema, 2001). Sous cer-

2. *Uniform Theta Assignement Hypothesis* ou assignation de rôles thématiques par le verbe conjugué.

taines conditions, le *parsing* obtenu est polynomial, contrairement à d'autres formalismes comme LFG ou HPSG qui peuvent apparaître plus intuitifs mais dont le *parsing* est exponentiel voire indécidable (à cause de la f-structure). Il apparaît alors pertinent d'explorer les approches basées sur le minimalisme.

Une critique importante des MG reste sur le versant de la représentation sémantique, qui n'a pas été travaillée à l'origine. Les premières propositions de calcul sémantique se sont montrées particulièrement *ad hoc* (Kobele, 2006). Ces propositions restent éloignées des approches montagoviennes classiques qui elles se basent sur des cadres logiques. Dans ce cas, la représentation sémantique est obtenue très efficacement avec une complexité basse à partir de la dérivation syntaxique. La sémantique est synchronisée à la syntaxe grâce au λ-calcul selon la distinction introduite par Curry entre les niveaux tectogrammaticaux (syntaxe abstraite) et phénogrammaticaux (réalisation de surface). On retrouve cette approche dans (Morrill, 1994), (de Groote, 2001), (Muskens, 2003), (Pollard, 2007).

Une version des MG dans un cadre logique a naturellement été proposée (Lecomte et Retoré, 2001) et (Lecomte, 2005). La notion d'analyse (*parsing*) devient la recherche d'une preuve dont la conclusion est la catégorie acceptante (sans hypothèse) et à laquelle on peut associer la suite de mots. Cette preuve est construite à partir des entrées lexicales associées aux mots de l'énoncé. Bien que l'intuition puisse laisser croire que cela complexifie l'analyse, la complexité algorithmique reste polynomiale. La nécessité de mobiliser simultanément la commutativité et la non-commutativité, mais sans leur donner d'interprétation linguistique, a introduit les grammaires minimalistes catégorielles (MCG) (Amblard, 2007) et (Amblard *et al.*, 2010) qui permettent une modélisation linguistique plus fine. Enfin, la classe de langages reconnue par les MCG contient celle reconnue par les MG (Amblard, 2011). Par ailleurs, (Michaelis, 2001) montre que les MG reconnaissent les langages faiblement sensibles au contexte (*midly context sensitive*), classe supposée des langues naturelles.

Le lien entre logique et théorie générative est réalisé par des preuves d'une restriction de la logique mixte (Retoré, 2004). Les preuves sont interprétées comme des représentations des relations syntaxiques, au sens de la théorie générative. Afin de poser le vocabulaire que nous utilisons, nous prenons l'exemple suivant[3] où l'on cherche à construire une preuve d'un syntagme nominal à partir d'une règle binaire :

$$\frac{\Delta \vdash un : det \quad \Gamma \vdash livre : nom}{\Delta; \Gamma \vdash un\ livre : SN}\ [\text{règle}]$$

Ici, nous avons une preuve, obtenue à partir de deux prémisses $\Delta \vdash un : det$ et $\Gamma \vdash livre : nom$, qui conduit à la conclusion $\Delta; \Gamma \vdash un\ livre : SN$ (en dessous du trait horizontal). Chaque élément de cette preuve est un séquent, composé en partie gauche d'un multi-ensemble d'hypothèses munies d'un ordre partiel série parallèle et en partie droite d'une conclusion, séparés par le symbole $\vdash$. Cette conclusion se

3. Cette dérivation ne respecte pas les MCG mais est proposée à titre d'illustration.

décompose en une chaîne de catactères et une formule. Ce sont les relations entre les hypothèses (comparables / incomparables) en partie gauche des séquents et les conclusions des deux premisses qui permettent d'appliquer le schéma de la règle qui explicite : comment les chaînes de caractères (qui peuvent être vues comme des traits) sont combinées, l'impact sur les ensembles d'hypothèses et la formule résultat.

La logique mixte est une extension du calcul de Lambek contenant simultanément les opérateurs commutatifs et non-commutatifs (*i.e.* introduction et élimination des connecteurs implicatifs et des tenseurs). Pour gérer dans le même calcul les différentes relations entre les hypothèses, une règle d'entropie (transformation de l'ordre par une restriction) est ajoutée. Par ailleurs, (Amblard et Retoré, 2014) ont montré la normalisation faible de ce calcul. Dans (Amblard, 2007) la définition des MCG est donnée à partir d'une composition d'un sous-ensemble des règles de la logique mixte. À partir de ces définitions, la normalisation est forte, ce qui permet de produire des analyses normalisées qui sont interprétées comme des représentations syntaxiques.

1.3. *Positionnement de la proposition*

La théorie générative est bien implantée dans la communauté linguistique, ce qui permet de s'appuyer sur une large littérature couvrant de nombreuses langues de différentes natures. De plus, l'un des arguments de Chomsky est de s'inspirer de la compréhension humaine pour proposer des analyses automatiques représentatives d'un processus cognitif. Les premières modélisations ont permis d'étudier l'expressivité du formalisme et l'efficacité de son *parsing*. Les MG surgénèrent, atteignant la classe des langages contextuels mais sa complexité est particulièrement intéressante car en bornant la taille des entrées (la taille des items lexicaux pour une grammaire donnée), la complexité redevient polynomiale, (Michaelis, 2001). Nous cherchons à installer ces propositions dans un cadre logique pour y synchroniser un calcul sémantique, via le λ-calcul.

Si la commutativité n'est pas linguistiquement motivée dans les MCG, la modélisation des *phases* la nécessite au delà des aspects techniques, argumentant de facto pour une telle logique. Nous revenenons sur les relations commutatives et non-commutatives des MCGs pour rendre compte de la notion de phase. Après avoir exposé les motivations et les notions mobilisées (théorie générative, phase, preuve en logique), nous définissons les MCG dans la section 2. Pour proposer un exemple, nous introduisons l'interface sur les chaînes de caractères. Dans la section 3, nous motivons l'utilisation des *phases* pour l'analyse et nous définissons les règles permettant d'en rendre compte. Puis nous revenons sur le calcul sémantique dans la section 4. La section 5 présente la dérivation d'une phrase simple, et revient sur l'analyse de cas particuliers. Enfin la dernière section conclut et ouvre sur les perspectives.

2. Grammaires Minimalistes Categorielles (MCG)

Les grammaires minimalistes catégorielles sont basées sur un fragment de la logique partiellement commutative (PCL) que nous appelons logique minimaliste (ML). Le calcul principal produit une preuve, comme nous l'avons introduit dans la section 1.2, dont la conclusion est un séquent vidé de ses hypothèses en partie gauche et ayant la catégorie acceptante (notée c) en partie droite. Sur cette preuve, nous synchronisons un calcul représentant la séquence de mots et un calcul sémantique (une formule logique produite par le λ-calcul). Dans cette section nous introduisons PCL, le calcul sur les chaînes de caractères et les MCG (lexique et règles). Le calcul sémantique étant profondément modifié par les *phases*, il est présenté dans la section 4.

2.1. *Partially Commutative Logic (PCL)*

PCL est la logique introduite dans (de Groote, 1996), (Ruet et Fages, 1998). Elle a été reprise dans (Retoré, 2004) dans une version se concentrant sur les réalisations et que nous utilisons. C'est une superposition du calcul de Lambek (*Intuitionistic Non-Commutative Multiplicative Linear Logic*) et de la logique linéaire intuitioniste multiplicative commutative (*Intuitionistic Commutative Multiplicative Linear Logic*).

Le calcul est basé sur les connecteurs non-commutatifs du calcul de Lambek ($\odot$, $\backslash$ et $/$) et des connecteurs commutatifs multiplicatifs et linéaires ($\otimes$ et $\multimap$). Pour chacun de ces connecteurs il existe une règle d'introduction et d'élimination.

L'un des avantages de cette logique est qu'elle est définie pour manipuler simultanément la commutativité et la non-commutativité, en particulier dans les contextes des séquents (en partie gauche). Ainsi, nous manipulons des multi-ensembles partiellement ordonnés d'hypothèses. La non-commutativité (le fait que deux hypothèses soient comparables) est notée ' ;' et la commutativité (le fait que deux hypothèses soient incomparables) est notée ','. Par exemple $A, B \vdash N$ signifie que pour obtenir un élément de type N, il nous faut deux hypothèses A et B, alors que pour $A; B \vdash N$, il faudra d'abord une hypothèse A avant une hypothèse B. Pour relâcher cet ordre, une règle d'*entropie* définie comme la substitution d'un ordre ' ;' par un ordre ',' est introduite (notée $\sqsubset$).

Nous faisons l'hypothèse que le calcul des MCG consomme les ressources introduites lexicalement. La ML est uniquement composée des règles d'élimination (privée de $\multimap_e$), de la règle d'axiome et de la règle d'entropie de PCL (voir la figure 2). Par exemple, la règle d'élimination de $\backslash$ s'écrit :

$$\frac{\Gamma \vdash A \quad \Delta \vdash A\backslash C}{\Gamma; \Delta \vdash C}\,[\backslash_e]$$

Cette règle est appliquée à partir de deux séquents, l'un ayant pour conclusion un A, l'autre ayant pour conclusion un $A\backslash C$. Ce dernier doit se trouver sur la droite du

$$\frac{\Gamma \vdash A \quad \Delta \vdash A\backslash C}{\Gamma;\Delta \vdash C}\,[\backslash_e] \qquad \frac{\Delta \vdash A/C \quad \Gamma \vdash A}{\Delta;\Gamma \vdash C}\,[/_e]$$

$$\frac{\Delta \vdash A \odot B \quad \Gamma, A; B, \Gamma' \vdash C}{\Gamma, \Delta, \Gamma' \vdash C}\,[\odot_e] \qquad \frac{\Delta \vdash A \otimes B \quad \Gamma, (A, B), \Gamma' \vdash C}{\Gamma, \Delta, \Gamma' \vdash C}\,[\otimes_e]$$

$$\frac{}{A \vdash A}\,[axiome] \qquad \frac{\Gamma \vdash C}{\Gamma' \vdash C}\,[\sqsubset], \Gamma' \sqsubset \Gamma$$

Figure 2. *Règles de la logique minimaliste (ML) utilisées pour les MCG.*

premier. La conclusion de la preuve construit le séquent ayant pour conclusion C (où l'on a éliminé le A sur la gauche). Les deux séquents sont réunis par une règle d'un connecteur non commutatif qui ordonne strictement les deux multi ensemble d'hypothèses (Γ sur la gauche et Δ sur la droite) et comparables (' ;'). Ainsi les hypothèses de Δ ne pourront pas commuter devant celles de Γ.

Les règles de tenseurs fonctionnent différemment. Elles combinent une preuve ayant un tenseur en conclusion avec une preuve ayant ces hypothèses (avec un ordre commutatif pour $\odot$ et non commutatif pour $\otimes$) dans la partie gauche du séquent. La règle substitue dans la position des hypothèses A, B les hypothèses du séquent ayant pour conclusion $A \otimes B$.

$$\frac{\Delta \vdash A \otimes B \quad \Gamma, (A, B), \Gamma' \vdash C}{\Gamma, \Delta, \Gamma' \vdash C}\,[\otimes_e]$$

2.2. *Étiquettes et preuves étiquetées (encodage de l'ordre des mots)*

Comme nous l'avons évoqué, on définit un calcul sur les chaînes de caractères. Nous ajoutons aux preuves de PCL les notions d'étiquettes et d'étiquetages, ainsi que les règles de manipulation de ces dernières. Ces définitions ont été entièrement introduites dans (Amblard, 2011). Nous ne reprenons ici que les définitions nécessaires.

Nous utilisons un ensemble de chaînes de caractères C et un ensemble disjoint V de variables ($C \cap V = \emptyset$). On note T l'union de ces deux ensembles ($T = C \cup V$). Comme nous nous situons dans le cadre de la théorie générative, nous faisons la différence entre la forme issue de la position de tête, celle de spécifieur et celle de complément, comme abordé dans la section 1.1. Nous manipulons des triplets où chaque position correspond respectivement aux positions spécifieur / tête / complément.

Définition 1 *L'ensemble des étiquettes Σ est l'ensemble des triplets de T^*. Une étiquette est considérée comme étant de la forme $r = (r_s, r_h, r_c)$. Pour la suite, nous notons $r_{-head} = (r_s, \epsilon, r_c)$ le triplet privé de son contenu en position de tête.*

L'étiquette pour l'élément lexical *enfant* est $(\epsilon, enfant, \epsilon)$. Elle associe en position principale (tête) la chaîne de caractères *enfant* et remplit les autres positions avec la chaîne de caractères vide (ϵ). De manière similaire, l'étiquette obtenue pour [lit [un livre]] est : $(\epsilon, lit, un\ livre)$ où à la position principale est associée la chaîne de caractères *lit* et en position de complément la chaîne de caractères *un livre*.

Ces étiquettes ne sont pas seulement composées de mots, mais également de variables, ce qui implique de définir une opération de substitution, ainsi qu'une notion de renommage (assurant que deux étiquettes utilisent des noms de variables différents).

Définition 2 *Une* substitution *est une fonction partielle de V dans T^*. Pour σ une substitution finie sur $x_1, \ldots, x_n$, où $\sigma(x_i) = t_i$, σ est dénoté par $[t_1/x_1, \ldots, t_n/x_n]$. Soient s une chaîne de caractères de T^* et r une étiquette, on note respectivement $s[t_1/x_1, \ldots, t_n/x_n]$ et $r[t_1/x_1, \ldots, t_n/x_n]$, la chaîne de caractères et l'étiquette obtenues par la substitution simultanée dans s ou r des variables par les valeurs associées dans σ (les variables non modifiées par σ restent les mêmes).*

Par exemple, pour $(x, lit, un\ livre)$ on peut substituer une chaîne de caractères à la variable x : $(x, lit, un\ livre)[l-\ enfant/x] = (l-\ enfant, lit, un\ livre)$

Définition 3 *Deux étiquettes r_1 et r_2 (respectivement deux chaînes de caractères s_1 et s_2) sont égales modulo un renommage, si $\exists\ \sigma$ tq $r_1.\sigma = r_2$ (resp. $s_1.\sigma = s_2$).*

La notion de renommage nous permet de considérer que $\exists\sigma$ tq $(x, lit, un\ livre).\sigma = (y, lit, un\ livre)$. Ainsi, lorsque des substitutions doivent être réalisées, il convient simplement de s'assurer que les noms de variables utilisées sont différents. Si ce n'est pas le cas, un simple renommage garantit la correction.

Définition 4 *L'opération de concaténation sur les chaînes de caractères, est notée $\bullet$.*

La concaténation sur les étiquettes est la concaténation des composantes une à une : $\forall r_1, r_2 \in \Sigma, r_1 \bullet r_2 = (r_{1s} \bullet r_{2s}, r_{1h} \bullet r_{2h}, r_{1c} \bullet r_{2c})$.

$Concat$ est l'opération de concaténation des composantes d'une étiquette dans leur ordre linéaire : $\forall r \in \Sigma, Concat(r) = r_s \bullet r_h \bullet r_c$

De manière intuitive, cette opération est nécessaire en fin de dérivation pour, à partir de $(l-\ enfant, lit, un\ livre)$, obtenir la forme finale *l'enfant lit un livre*.

Ces triplets et ces opérations permettent de définir la notion de preuves étiquetées.

Définition 5 *$\forall\ (r_s, r_t, r_c) \in \Sigma$, un G-*séquent *est un séquent tq $\Gamma \vdash_G (r_s, r_t, r_c) : B$ où Γ est un ensemble d'hypothèses partiellement ordonné par un ordre série-parallèle.*

Définition 6 *Un G-*étiquetage *est une dérivation d'un G-sequent obtenu par :*

$$\frac{\Gamma \vdash_G r_1 : A \,/\, B \quad \Delta \vdash_G r_2 : B}{\langle \Gamma; \Delta \rangle \vdash_G (r_{1s}, r_{1t}, r_{1c} \bullet Concat(r_2)) : A}\,[/_e] \qquad \frac{\langle s, A \rangle \in Lex}{\vdash_G (\epsilon, s, \epsilon) : A}\,[Lex]$$

$$\frac{\Delta \vdash_G r_2 : B \quad \Gamma \vdash_G r_1 : B \setminus A}{\langle \Gamma; \Delta \rangle \vdash_G (Concat(r_2) \bullet r_{1s}, r_{1t}, r_{1c}) : A}\,[\setminus_e] \qquad \frac{x \in V}{x : A \vdash_G (\epsilon, x, \epsilon) : A}\,[axiome]$$

$$\frac{\Gamma \vdash_G r_1 : A \otimes B \quad \Delta[x : A, y : B] \vdash_G r_2 : C}{\Delta[\Gamma] \vdash_G r_2[Concat(r_1)/x, \epsilon/y] : C}\,[\otimes_e] \qquad \frac{\Gamma \vdash_G r : A}{\Gamma' \vdash_G r : A}\,[\sqsubset], \Gamma' \sqsubset \Gamma$$

$$\frac{\Gamma \vdash_G r_1 : A \odot B \quad \Delta[x : A; y : B] \vdash_G r_2 : C}{\Delta[\Gamma] \vdash_G r_2 \bullet r_1 : C}\,[\odot_e] \qquad \text{où } Var(r_1) \cap Var(r_2) = \emptyset$$

La règle Lex permet d'introduire dans une dérivation un élément issu du lexique et la règle d'$axiome$ d'introduire des hypothèses dans la dérivation.[4] $/$ et $\setminus$ utilisent des concaténations prenant en compte les relations gauche/droite (spécifieur/complément) pour introduire une étiquette dans une autre. $\otimes$ utilise la règle de substitution sur les étiquettes et $\odot$ la règle de concaténation entre deux étiquettes. Ces règles sont utilisées pour définir les règles des MCG et seront illustrées dans la section 2.4.

2.3. *Lexique*

Une grande partie de l'information nécessaire pour l'analyse dans les MCG est lexicalisée. Le lexique fait correspondre à chaque mot un séquent étiqueté. L'étiquette contient la forme du mot en position de tête du triplet, ainsi que deux chaînes vides. La partie gauche du séquent peut contenir un multi-ensemble d'hypothèses. La formule associée en partie droite du séquent respecte une forme prédéfinie : elle est construite à partir d'une série $\otimes$ ou de $\odot$ et d'une formule de base, encadrée par des $/$ et $\setminus$. Un exemple d'item lexical est $d, k; h \vdash (\epsilon, mot, \epsilon) : (c_m \setminus \ldots \setminus c_1 \setminus (b_1 \otimes \ldots \otimes b_n \otimes a))/d$.

La forme induite par cette définition est le pendant des structures de traits utilisées dans les MG de Stabler. Les $/$ et $\setminus$ permettent de combiner les preuves entre elles. On peut voir cette partie comme celle construisant la preuve (au contraire de la suivante qui modifie sa structure). La formule se termine soit par une formule de base correspondant à la catégorie du syntagme, soit par une séquence de $\odot$ ou de $\otimes$. Si les formes résultantes peuvent paraître complexes, elles restent isomorphes à celles utilisées dans les MG (Amblard, 2007).

4. Nous reviendrons en détails sur la notion d'hypothèse en définissant la règle de déplacement.

2.4. *Règles des MCG*

Le MP est basé sur deux règles : la fusion (*merge*) qui combine deux dérivations et le déplacement (*move*) qui redéfinit la structure interne de la dérivation (en opérant un déplacement explicite). Nous formalisons ces deux principes grâce aux règles de la ML. La règle de fusion (*merge*) utilise l'élimination de $/$ ou $\backslash$, opérations que l'on retrouve dans les différentes versions issues des grammaires catégorielles (Lambek, 1958), (Steedman, 1987) et (Moortgat, 1997). La règle de déplacement (*move*) est simulée très différemment. Il s'agit dans un premier temps d'introduire des hypothèses (en partie gauche du séquent) ayant pour rôle de marquer des positions, puis lorsque toutes les positions sont présentes d'y substituer un syntagme grâce à l'élimination d'un tenseur. Le déplacement n'est pas explicite d'une position A vers une position B, mais lorsque les deux positions sont marquées, on y substitue directement un contenu.

Dans les MCG, la ***fusion*** est la règle qui combine deux preuves. Pour rendre correctement compte de l'ordre des mots, elle utilise la non-commutativité. Mais dans la même application, la fusion combine également les hypothèses des deux prémisses. D'un point de vue linguistique, les hypothèses fonctionnent au même niveau (à l'intérieur d'un même constituant). Elles doivent pouvoir commuter pour que leur accessibilité soit équivalente, ce que la non-commutativité précédente ne permet pas.

Pour réaliser ces opérations dans les MCG, la fusion est une élimination de $/$ ou $\backslash$ immédiatement suivie de l'application de la règle d'entropie (qui réduit l'ordre non-commutatif en commutatif). La fusion se décline en deux versions dépendant du déclencheur gauche ou droite, soit ici de la formule portant le connecteur $/$ ou $\backslash$. La règle reprenant ces deux applications est notée *mg* (pour *merge*). Pour le calcul sur les chaînes de caractères, la fusion est la concaténation d'une étiquette dans une autre (en fonction de la position gauche / droite).

Déclencheur gauche :

$$\dfrac{\dfrac{\Delta \vdash s : B \qquad \Gamma \vdash (r_s, r_h, r_c) : B \setminus A}{\Delta; \Gamma \vdash (Concat(s) \bullet r_s, r_h, r_c) : A}\,[\setminus_e]}{\Delta, \Gamma \vdash (Concat(s) \bullet r_s, r_h, r_c) : A}\,[\sqsubset] \Longrightarrow \dfrac{\Delta \vdash s : B \quad \Gamma \vdash (r_s, r_h, r_c) : B \setminus A}{\Delta, \Gamma \vdash (Concat(s) \bullet r_s, r_h, r_c) : A}\,[mg]$$

Déclencheur droit :

$$\dfrac{\dfrac{\Gamma \vdash (r_s, r_h, r_c) : A \,/\, B \qquad \Delta \vdash s : B}{\Gamma; \Delta \vdash (r_s, r_h, r_c \bullet Concat(s)) : A}\,[/_e]}{\Gamma, \Delta \vdash (r_s, r_h, r_c \bullet Concat(s)) : A}\,[\sqsubset] \Longrightarrow \dfrac{\Gamma \vdash (r_s, r_h, r_c) : A \,/\, B \quad \Delta \vdash s : B}{\Gamma, \Delta \vdash (r_s, r_h, r_c \bullet Concat(s)) : A}\,[mg]$$

En supposant que l'on ait les séquents $\vdash (\epsilon, un, \epsilon) : k \otimes d \,/\, n$ pour le déterminant et $\vdash (\epsilon, livre, \epsilon) : n$ pour le nom. Ici, la tête du constituant est le déterminant, comme dans la théorie générative, ainsi que dans la plupart des formalismes basés sur les grammaires catégorielles. On les combine par une opération $[mg]$:

$$\frac{\vdash (\epsilon, un, \epsilon) : (k \otimes d) \,/\, n \quad \vdash (\epsilon, livre, \epsilon) : n}{\vdash (\epsilon, un, livre) : k \otimes d}\,[mg]$$

Pour rendre compte de (Stabler, 2001), extensions des MG, la fusion est enrichie de règles permettant de modifier les positions des chaînes de caractères dans les étiquettes. Afin de réduire le nombre de notations, elles sont encore notées *mg*. Les connecteurs sont décorés des symboles $<$ et $>$. On distingue deux sortes de règles :

Head movement où la tête de l'élément combiné est concaténée dans la tête de l'élément résultant. Les autres composantes du triplet sont concaténées comme dans la règle de fusion. Les $<$ et $>$ pointent vers le connecteur. À partir de cette idée on peut inférer quatre règles : deux pour distinguer l'ordre dans la concaténation entre les deux têtes, et deux autres qui prennent en compte les déclencheurs à gauche ou à droite. Voici l'une des règles, les autres pouvant être déduites :

$$\frac{\Gamma \vdash (r_{spec}, r_{tete}, r_{comp}) : A /_{<} B \qquad \Delta \vdash s : B}{\Gamma, \Delta \vdash (r_{spec}, r_{tete} \bullet s_{tete}, r_{comp} \bullet Concat(s_{-tete})) : A}\,[mg]$$

Affix hopping où la tête du déclencheur est concaténée avec la tête de l'élément combiné. Les autres composantes du triplet sont concaténées comme dans la règle de fusion. Les $<$ et $>$ pointent à l'extérieur du connecteur. De la même manière, il existe quatre versions de la règle pour distinguer la gauche de la droite, et le statut du déclencheur. Également à titre d'exemple, voici l'une des règles :

$$\frac{\Gamma \vdash (r_{spec}, r_{tete}, r_{comp}) : A /_{>} B \qquad \Delta \vdash s : B}{\Gamma, \Delta \vdash (r_{spec}, \epsilon, r_{comp} \bullet Concat((s_{spec}, r_{tete} \bullet s_{tete}, s_{comp}))) : A}\,[mg]$$

L'encodage du ***déplacement*** est structurellement différent. Les hypothèses sont envisagées comme des ressources marquant les traits relatifs à un syntagme. Lorsqu'un multi-ensemble (partie gauche) d'un séquent possède les hypothèses avec l'ordre adéquat, on réalise une substitution qui simule un déplacement de la position de la première hypothèse vers la position de la seconde. Nous ne réinterprétons pas littéralement des relations internes comme dans les grammaires minimalistes, mais nous construisons une dérivation possédant des hypothèses en partie gauche des séquents marquant des positions spécifiques. De cette manière, un déplacement est le déchargement d'hypothèses incomparables (qui respectent l'ordre commutatif) par un $\otimes$.

Revenons sur la notion d'hypothèses de nos séquents. La séquence d'hypothèses de la preuve contient exactement la séquence de ressources disponibles pour la suite de la dérivation. Dans ce cas, cette séquence est un ensemble de ressources et non une liste. D'où la nécessité d'introduire la non-commutativité entre ces hypothèses pour ne pas présupposer un ordre canonique sur la suite d'opérations, ce qui explique l'application de la règle d'entropie dans la définition de la fusion. Par ailleurs nous supposons l'existence d'hypothèses lexicales basées sur la règle d'axiome. Pour le calcul sur les

chaînes de caractères, on lui associe un triplet contenant une variable. Ainsi pour une catégorie H, la dérivation pourra mobiliser un séquent appelé hypothèse lexicale :

$$H \vdash (\epsilon, x, \epsilon) : H$$

Pour les chaînes de caractères, le déplacement est une substitution. La variable la plus récemment entrée dans la dérivation est substituée par la concaténation de la structure *déplacée*, les autres positions reçoivent la chaîne vide (ϵ) qui correspond dans la réalisation de surface à la trace du déplacement. La règle est notée *mv* (pour *move*).

$$\frac{\Gamma \vdash r_1 : A \otimes B \quad \Delta[u : A, v : B] \vdash r_2 : C}{\Delta[\Gamma] \vdash r_2[Concat(r_1)/u, \epsilon/v] : C}\,[mv]$$

En supposant que la dérivation a produit d'un côté notre exemple précédent $\vdash (\epsilon, un, livre) : k \otimes d$ et d'un autre côté $k, d \vdash (v, lit, u) : V$ qui correspond au verbe *lire* ayant reçu deux hypothèses (d et k). Il est alors possible d'appliquer mv :

$$\frac{\vdash (\epsilon, un, livre) : k \otimes d \quad k, d \vdash (v, lit, u) : V}{\vdash (un\ livre, lit, \epsilon) : V}\,[mv]$$

Il ne faut pas considérer ici que l'analyse est terminée et que l'ordre entre les mots n'est pas valide. La suite de la dérivation fait évoluer la catégorie du verbe et la chaîne de caractères lui correspondant prend d'autres positions. Les triplets distinguant les positions spécifieur / tête / complément permettent ce type de manipulation.

3. Encodage des Phases dans les MCG

Comme nous l'avons évoqué dans la section 1.1, Chomsky propose dans (Chomsky, 1999) que l'analyse soit réalisée à partir de *phases*. Cette vision permet de définir des îlots (syntaxiques) et de se rapprocher du processus humain d'analyse et de compréhension. Dans la théorie générative, comme dans de nombreuses théories linguistiques, l'analyse se réalise autour du verbe. Ainsi les *phases* correspondent à des configurations spécifiques du verbe. Chomsky en identifie au moins deux : le verbe avant qu'il ne reçoive son inflexion noté VP, et la phrase pouvant être considérée comme une complétive notée cP (en fin d'analyse pour une phrase simple). Le verbe ayant recu son inflexion est noté tP (pour time) et le verbe ayant reçu ses premiers arguments est noté vP.

Si les *phases* sont définies à partir de modifications du syntagme verbal, toutes celles qu'il reçoit ne sont pas des *phases*. Le moment où le syntagme verbal reçoit son inflexion tP, ou encore celui où il rencontre ses différents arguments vP n'en sont pas. La figure 3 schématise les différentes catégories du noyau verbal sous forme d'arbre syntaxique. L'analyse que nous proposons distingue l'évolution du verbe par des éléments lexicaux (comme l'inflexion), des autres parties que sont les *phases*.

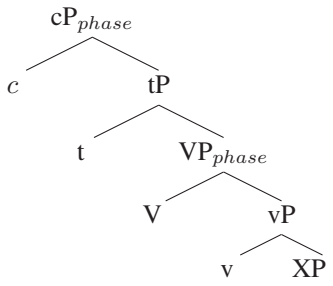

Figure 3. *Structure de l'analyse d'une phrase simple : position des deux phases dans le noyau verbal cP et VP.*

Dans la définition de phase, Chomsky suppose qu'une partie des déplacements est réalisée. Ils interviennent à la fin de la phase, dans ce que l'on appelle le *transfert*. Après ce dernier, les ressources encore présentes en position de complément ne sont plus accessibles au reste de l'analyse, ce qui définit un îlot (*island*) au sein de l'analyse et une condition de non accessibilité, la *Phase Impenetrability Condition* (PIC). La traduction dans notre formalisme suppose qu'une phase bloque l'accessibilité d'une partie de la séquence d'hypothèses. Nous supposons qu'il s'agit d'hypothèses comparables (entretenant un ordre non-commutatif), en partie gauche du séquent. Ainsi, la non-commutativité se comporte comme une frontière entre les hypothèses. Cet argument donne un sens linguistique à l'utilisation de la commutativité (nécessaire pour la fusion) et de la non-commutativité (nécessaire pour la *phase*).

Pour une phrase simple affirmative à l'actif, le noyau verbal se décompose en 4 catégories (cP, tP, VP, vP), voir figure 3. Pour chacune, on introduit un item lexical qui intervient dans la dérivation soit directement (par une fusion), soit par une *phase* :

vP l'item qui introduit le verbe

VP Mode : donne le cas accusatif à l'objet et demande un sujet

tP $\mathrm{Inflection}$: apporte l'inflection au verbe[5], et le cas syntaxique nominatif

cP Comp : vérifie si la dérivation est interprétée en position de complément (cette partie est mobilisée pour les questions, l'insertion dans une clause relative, etc.)

Les items lexicaux de mode et comp correspondent aux déclencheurs des phases et les deux autres sont relatifs à l'évolution de l'analyse du syntagme verbal.

La *phase* a un double effet, elle combine deux preuves pour former une seule dérivation, et elle déclenche une succession de déplacements (*transfert*). Pour modéliser

5. Dans la suite nous nous limitons à des exemples où le verbe est introduit directement avec sa forme flexionnelle afin de simplifier la présentation des chaînes de caractères.

les *phases* nous nous appuyons sur l'idée qu'elles marquent des points dans la dérivation qui rendent inaccessibles certaines ressources. Ici, les ressources sont stockées sous la forme d'un multi-ensemble d'hypothèses (partie gauche du séquent). Nous utilisons dans ce multi-ensemble deux hypothèses H_1 et H_2, comparables (séparées par un ' ;' marquant la non-commutativité) : $\Delta_1, H_1; H_2, \Delta_2 \vdash A$.

Comme la règle d'entropie ne peut être mobilisée que dans la règle de fusion, cet ordre ' ;' est conservé jusqu'à la fin de la dérivation. On interprète alors ce point de comparaison (ordre non-commutatif) comme le point de la *phase*. Son introduction bloque l'accès à une partie du contenu, ce qui empêche dans la suite de la dérivation de réaliser des déplacements à partir d'hypothèses en position de complément. D'un autre côté, ces hypothèses doivent pouvoir être déchargées, avec la règle $\odot_e$ de la ML. Une *phase* est alors le déchargement d'hypothèses comparables (entretenant un ordre non-commutatif), qui sont introduites par les items lexicaux.

Définition 7 *La règle de phase se décompose en deux parties :*

1) le déchargement d'hypothèses comparables, notée $[phase]$

$$\frac{\Delta_s, \Delta_h, \Delta_c \vdash (s_s, s_h, s_c) : X \odot Y \quad \Gamma_s, X; Y, \Gamma_c \vdash (r_s, r_h, r_c) : Z}{\Gamma_s, \Delta_s, \Delta_h \vdash (r_s \bullet s_s, r_h, s_h \bullet s_c \bullet r_c) : Z}\,[phase]$$

2) une partie transfert qui réalise tous les déplacements possibles, notée $[phase_t]$. *Pour cela la règle* $[mv]$ *est utilisée.*

La règle de phase est donc composée d'une élimination de tenseur non-commutatif, puis de l'application de [mv] (potentiellement plusieurs fois). Une condition est ajoutée sur Δ_c *et* Γ_c *qui doivent être vides après* $[phase_t]$ *car le contenu interne d'une phase n'est plus accessible en dehors de cette dernière*[6]*. Les phases contrôlent l'analyse syntaxique grâce à cette condition implémentée dans la règle.*

Finalement, pour les mêmes argument que pour la règle de fusion, nous étendons la règle de la *phase* aux notions de *head-mouvement* et d'*affix-hopping*. À nouveau nous utilisons les mêmes marqueurs avec les mêmes effets. À titre d'exemple l'une des instances, où la tête du triplet est la concaténation des deux têtes, est :

$$\frac{\Delta_s, \Delta_h, \Delta_c \vdash (s_s, s_h, s_c) : X \odot_< Y \quad \Gamma_s, X; Y, \Gamma_c \vdash (r_s, r_h, r_c) : Z}{\Gamma_s\Delta_s, \Delta_h \vdash (r_s \bullet s_s, s_h \bullet r_h, s_c \bullet r_c) : Z}\,[phase]$$

Les MCG sont donc composées de la règle $[mg]$, ainsi que de ses différentes variantes, la règle $[mv]$ et la règle de $[phase]$. On note MCG$_{phase}$ les MCG augmentées de la règle de *phase*.

Nous définissons les items lexicaux des deux *phases* :

6. ou dit autrement, une dérivation voyant l'une de ses phases ne respectant pas cette condition ne pourra jamais aboutir à une dérivation acceptante (vidée de toutes ses hypothèses).

– mode : $V; v \vdash k \setminus d \setminus V$

La partie gauche du séquent peut se lire comme "à partir d'un v on produit un V", sous entendu par la réalisation de la *phase*. La partie droite du séquent exprime que la dérivation doit être combinée avec un cas syntaxique et une catégorie d, ce qui correspond à la définition de *mode* présentée page 14.

– comp : $c; t \vdash c$

Dans cet exemple, cette *phase* est porteuse de moins d'information. On peut lire la partie gauche du séquent comme "à partir d'un t, on dérive un c" et la partie droite apporte la catégorie acceptante pour la dérivation. Il ne faut pas interpréter que son effet est nul, puisque la partie de transfert de la *phase* doit être réalisée pour vider intégralement la partie gauche du séquent et produire une dérivation acceptante.

Par ailleurs, les items lexicaux associés au verbe et à l'inflexion comportent eux aussi une information relative à la *phase*. Ils sont définis par rapport à la phase jusqu'à laquelle ils peuvent agir. Enfin, ces opérations sont déclenchées par un $\odot_<$ permettant de conserver en position de tête la chaîne de caractères correspondant au verbe.

– verbe : $\vdash (V \odot_< v) \,/\, d$

Un verbe est combiné avec une catégorie d, utilisée pour la catégorie de l'objet syntaxique, avant d'atteindre la première *phase*.

– inflection : $\vdash k \setminus (c \odot_< t) \,/\, V$

L'inflexion est combinée à une dérivation ayant la catégorie V, et permet ainsi de poursuivre en intégrant un cas syntaxique (nominatif).

Pour illustrer une dérivation à partir des *phases*, nous revenons sur les premières étapes de la dérivation de l'exemple (1)[7]. Nous utilisons l'exemple (3) contenant un seul déterminant (pour simplifier la présentation du calcul sémantique). La dérivation complète est présentée dans la figure 4 de la section 5 avec le calcul sémantique.

(3) Un enfant lit un livre

verbe	$\vdash (\epsilon, lit, \epsilon) : \quad (V \odot_< v) \,/\, d$
mode	$V; v \vdash (\epsilon, \epsilon, \epsilon) : \quad k \setminus d \setminus V$

Première étape : item lexical du verbe

La première partie de la dérivation est la saturation de la première position du verbe par une hypothèse lexicale $d \vdash (\epsilon, v, \epsilon) : d$. Cette hypothèse correspond à la position principale du constituant qui occupe la position d'objet dans la phrase. Ce constituant n'est pas directement inséré dans la dérivation car tous ses traits ne sont pas représentés par des hypothèses (il manque la position de l'attribution de cas k).

7. Il s'agit de montrer les mécanismes en œuvre et non pas un exemple de la complexité maximale reconnue. Pour prendre en compte des phénomènes syntaxiques plus élaborés, il est nécessaire de définir de nouveaux items lexicaux et non de redéfinir le formalisme.

$$\frac{\vdash (\epsilon, lit, \epsilon) : (V \odot_{<} v) \,/\, d \quad d \vdash (\epsilon, u, \epsilon) : d}{d \vdash (\epsilon, lit, u) : (V \odot v)}\,[mg]$$

La dérivation autour du syntagme verbal peut maintenant être combinée avec une *phase*. Une manière de comprendre ces preuves est d'interpréter la non-commutativité entre V et v comme la saturation de toutes les hypothèses utilisées dans la construction d'un syntagme de catégorie v pour produire un syntagme V.

Deuxième étape : item lexical de mode

Nous trouvons ces hypothèses dans la preuve induite par la deuxième entrée du verbe : mode. Nous fusionnons les différentes positions de l'entrée mode avec des hypothèses lexicales (de type k et de type d) :

$$\frac{d \vdash (\epsilon, w, \epsilon) : d \qquad \dfrac{k \vdash (\epsilon, v, \epsilon) : k \quad V; v \vdash (\epsilon, \epsilon, \epsilon) : k \setminus d \setminus V}{k, V; v \vdash (v, \epsilon, \epsilon) : d \setminus V}\,[mg]}{d, k, V; v \vdash (w\ v, \epsilon, \epsilon) : V}\,[mg]$$

À ce point de la dérivation, le résultat est combiné avec le résultat de la première étape par une opération de $[phase]$. Les composantes de l'étiquette de la preuve de gauche sont fusionnées avec celles de l'étiquette de la preuve de droite.

$$\frac{d \vdash (\epsilon, lit, u) : (V \odot_{<} v) \quad d, k, V; v \vdash (w\ v, \epsilon, \epsilon) : V}{d, k, d \vdash (w\ v, lit, u) : V}\,[phase]$$

La *phase* se poursuit par la réalisation de tous les déplacements possibles, ce qui permet d'introduire l'objet de la phrase. C'est à ce moment que le constituant objet apparaît véritablement dans la dérivation aux positions des hypothèses précédemment introduites. Le constituant objet est construit avec les items lexicaux de *un* et *livre* comme présenté dans la section 2.4. La dérivation choisit l'hypothèse d qui décharge entièrement la partie en position de complément de la *phase*. Le choix de l'autre d est bloqué par la PIC qui arrêterait immédiatement la dérivation.

$$\frac{\vdash (\epsilon, un, livre) : k \otimes d \quad d, k, d \vdash (w\ v, lit, u) : V}{d \vdash (w\ un\ livre, lit, \epsilon) : V}\,[phase_t]$$

Dans cette séquence, la non-commutativité contrôle la *phase*, puis la commutativité permet le déchargement des hypothèses du syntagme nominal. Les hypothèses apparaissent dans un ordre non prédéfini et leurs ordres contrôlent la dérivation.

4. Interface syntaxe-sémantique

Un autre argument en faveur de la prise en compte de la notion de *phase* provient de la définition du calcul de la sémantique. Pour ce calcul on associe un terme logique à chaque élément du lexique et on cherche à les composer. (Amblard, 2007)

et (Lecomte, 2008) introduisent un tel calcul pour les MCG (sans les *phases*). Ils supposent qu'aux différents termes de la construction du syntagme verbal correspondent les assignations de rôles thématiques, comme dans (Hale et Keyser, 1993.), (Kratzer, 1994) ou (Baker, 1997). Ainsi le terme du verbe est associé à son item lexical, et l'inflexion apportera un prédicat $agent$. Cette vision du calcul nécessite de suivre une position néo-davidsonienne de la sémantique. L'événement décrit est ainsi réifié (défini par une variable), ce qui permet de le décrire avec plusieurs prédicats. Cependant, le terme principal du verbe introduisant le prédicat, $read$ dans notre exemple, continue de contenir comme argument les variables mobilisées en syntaxe profonde. Par ailleurs, ce calcul nécessite d'interpréter les termes dans leur contexte (par exemple, le quantificateur de la variable d'événement est introduit seulement à la fin de la dérivation), d'où l'utilisation des continuations.

L'utilisation de la notion de *phase* permet de définir une interface simplifiée. Nous reprenons l'encodage des continuations introduite dans (de Groote, 2006). Nous utilisons le λ-calcul avec un type étendu aux contextes. L'interprétation du verbe transitif n'est plus $\lambda xy.verbe(y, x)$, comme dans la théorie montagovienne, mais $\lambda xye\phi.verbe(y, x) \wedge \phi e$. Ici e est la variable représentant le contexte gauche dans lequel le terme est utilisé (ce à partir de quoi il est interprété) et ϕ le contexte droit (ce vers quoi il poursuivra son interprétation). Ces idées permettent d'augmenter ce calcul à celui du discours, en étendant la notion de portée. (de Groote, 2006) propose de voir le contexte comme une liste de variables introduites et accessibles pour le reste de l'interprétation. L'opérateur ' : :' est utilisé comme constructeur de liste.

L'interface syntaxe-sémantique est ainsi très réduite : dans le calcul sémantique l'application fonctionnelle est associée à la fusion et au déplacement ; la *phase* utilise aussi l'application fonctionnelle ainsi qu'un terme spécifique qui décharge deux termes de leurs variables avant de les recombiner. Ce terme dépend du nombre d'arguments restant dans celui du verbe :

– 2 arguments : $\lambda T_1 T_2 OSr.S(\lambda x.O(\lambda ye\phi.T_1(\lambda A.A)(\lambda B.B)rxye(\lambda e'.T_2rye'\phi)))$
– 1 argument : $\lambda T_1 T_2 Sr.S(\lambda xe\phi.T_1(\lambda A.A)rxe(\lambda e'.T_2rxe'\phi)))$

Ces termes peuvent paraître complexes, mais ils ne font que redistribuer les variables sur leurs arguments, ce qui permet d'unifier les différentes variables (contextes, réification, etc.). C'est le seul niveau de complexité de cette interface. Les premières versions contenaient plusieurs éléments calculatoires *ad hoc*. Seuls les termes lexicalisés participent à la formule finale. Ainsi, l'interprétation sémantique des hypothèses lexicales mobilisées dans l'analyse est l'identité. Une illustration de l'utilisation de ce calcul est proposée dans la section suivante.

Les termes associés au déterminant et au nom restent traditionnels (augmentés de la notion de continuation), ceux associés aux items lexicaux portant les *phases* (mode et comp) apportent l'information sur les rôles thématiques correspondant au cas syntaxique qu'ils introduisent dans la dérivation, voir section suivante.

5. Dérivation d'une phrase simple

Avant de commenter la longue dérivation de l'exemple (1) présenté dans la figure 4, nous rassemblons les éléments du lexique que nous avons introduit au cours de l'article, ainsi que leur sémantique (Id dénote l'identité.) :

un	$\vdash (\epsilon, un, \epsilon) : k \otimes d \,/\, n$	$\lambda PQe\phi.\exists x.Px(x :: e)(\lambda e'.Qxe'\phi)$
$enfant$	$\vdash (\epsilon, enfant, \epsilon) : n$	$\lambda ze\phi.child(z) \wedge \phi e$
$livre$	$\vdash (\epsilon, livre, \epsilon) : n$	$\lambda ze\phi.book(z) \wedge \phi e$
$lire$	$\vdash (\epsilon, lit, \epsilon) : (V \odot_< v) \,/\, d$	$\lambda OSr.S\lambda x.O\lambda ye\phi.read(r, x, y) \wedge \phi e$
mode	$V; v \vdash (\epsilon, \epsilon, \epsilon) : k \setminus d \setminus V$	$\lambda rze\phi.patient(r, z) \wedge \phi e$
infl	$\vdash (\epsilon, -, \epsilon) : k \setminus (c \odot t) /_< V$	Id
comp	$c; t \vdash (\epsilon, \epsilon, \epsilon) : c$	$\lambda rze\phi.agent(r, z) \wedge \phi e$

La normalisation de PCL proposée dans (Amblard et Retoré, 2014) assure qu'il est possible de produire une dérivation acceptante ayant une forme normalisée.

Comme nous l'avons précédemment présenté, les groupes nominaux sont construits à partir de l'item lexical du déterminant $\vdash (k \otimes d) \,/\, n$ qui est combiné avec celui d'un nom $\vdash n$. Le résultat est un constituant de catégorie d (pour *determinal phrase*) auquel il manque l'assignation d'un cas syntaxique (k) ce qui s'écrit par la catégorie $k \otimes d$. Du point de vue sémantique, il s'agit de combiner l'introduction d'un quantificateur avec le prédicat du nom. Le terme du quantificateur distribue les arguments en les passant à la continuation du terme du nom.

Dans la section 3, nous avons présenté la première partie de la construction de la *phase* qui apparaît dans la première partie de la figure 4. Cette partie du calcul a permis de rassembler le verbe et son inflexion ainsi que son objet. Dans le calcul sémantique, le terme de la phase est composé d'un prédicat dénotant un rôle thématique, ainsi que d'une continuation. On lui applique l'identité jusqu'à l'application de la règle de *phase*. Elle mobilise un terme qui redistribue les variables sur le terme précédent et le terme associé au verbe, permettant de les unifier.

Cette *phase* se termine par le *transfert* qui réalise les déplacements. Il s'agit ici d'introduire le syntagme nominal correspondant à l'objet (*un livre*). La sémantique produit alors le terme $\lambda Sr.S(\lambda xe\phi.\exists z.book(z) \wedge read(r, x, z) \wedge patient(r, z) \wedge \phi(z :: e))$, rassemblant le prédicat du verbe, celui du nom et le prédicat signifiant que la variable de ce nom est en relation $patient$ dans l'événement en cours de description. La continuation contient en plus la variable introduite par le quantificateur.

La dérivation fusionne cette dérivation et l'item lexical de l'inflexion. L'utilisation d'un *head-movement* assure de concaténer la chaîne de caractères du verbe avec son inflexion. Puis, la dérivation est combinée avec une hypothèse de type k. Ces deux fusions ne modifient pas la sémantique car nous utilisons deux fois l'identité.

Cette dérivation produit un élément de type $c \odot_< t$ qui contient deux hypothèses k et d en partie gauche du séquent. Ces hypothèses permettront de faire entrer le syntagme nominal sujet dans la dérivation dans la partie de transfert. Nous rappelons

Figure 4. *Dérivation syntaxique (en noir), sur les chaînes de caractères (en bleu) et sémantique (en rouge) de l'exemple 1*

que la normalisation des MCG nous assure de travailler sur une forme normale de la preuve où les éliminations de tenseurs sont appliquées dès que la structure de la preuve le permet. Le résultat est combiné avec l'item lexical comp par la seconde *phase*. Sa sémantique introduit le second rôle thématique. La dérivation produit le séquent $k, d \vdash (z, lit, w\ un\ livre) : c$. Le calcul sémantique utilise le terme de la *phase* correspondant au nombre d'arguments restant dans le terme du verbe, c'est-à-dire un. La *phase* se termine, comme la dérivation, par le *transfert* qui introduit le syntagme nominal sujet dans les positions de ses variables. L'introduction de ce terme dans son contexte (le discours par exemple) permet de fixer la variable d'événement.

Il faut noter que les continuations stockent dans le contexte les variables introduites par les quantificateurs : $(z :: x :: e)$ exprime que les variables z et x sont ajoutées au contexte e, les rendant accessibles pour interpréter la suite du calcul sémantique.

Analyse d'une question : déplacement cyclique au travers de la phase

Pour illustrer le fonctionnement des *phases*, nous revenons sur les déplacements cycliques. L'enjeu est de faire passer de la position de complément à spécifieur les marqueurs d'un syntagme lors du *transfert*. Ce phénomène intervient dans la dérivation d'une question, comme dans l'exemple (2) où le dernier trait du syntagme objet est introduit par l'item lexical de comp : $c; t \vdash (\epsilon, \epsilon, \epsilon) : wh \setminus c$. Ce syntagme doit alors rester accessible après la réalisation de la première *phase*. Pour cela, on lie les deux premières hypothèses avec une hypothèse lexicale $k \otimes d \vdash (\epsilon, x, \epsilon) : k \otimes d$ qui sera déchargé de la dérivation avec le séquent de quel.

$\vdash (\epsilon, quel, \epsilon) : (wh \otimes (k \otimes d)) \,/\, n$

La dérivation est similaire jusqu'à la partie de *transfert* de la *phase*. Du point de vue sémantique, le déplacement n'ayant pas lieu, le déplacement cyclique doit inverser l'ordre d'application des variables.

$$\dfrac{\begin{array}{ll} k \otimes d \vdash (\epsilon, x, \epsilon) : k \otimes d & d, k, d \vdash (w\ v, lit, u) : V \\ \lambda PSO.POS & \lambda OSr.S(\lambda x.O(\lambda ye\phi.read(r, x, y) \\ & \wedge patient(r, y) \wedge \phi e)) \end{array}}{\begin{array}{l} d, k \otimes d \vdash (w\ x, lit, u) : V \\ \lambda SOr.S(\lambda x.O(\lambda ye\phi.read(r, x, y) \wedge patient(r, y) \wedge \phi e)) \end{array}}\ [phase_t]$$

La dérivation se poursuit jusqu'à l'utilisation de l'item lexical comp qui est combiné avec un séquent hypothèse avant de déclencher la première partie de la *phase*.

$$\dfrac{\begin{array}{ll} k, d, k \otimes d \vdash (z, lit, w\ x) : (c \odot_< t) & \dfrac{\begin{array}{ll} wh \vdash (\epsilon, y, \epsilon) : wh & c; t \vdash (\epsilon, \epsilon, \epsilon) : wh \setminus c \\ Id & \lambda re\phi.agent(r, z) \wedge \phi e \end{array}}{\begin{array}{l} wh, c; t \vdash (y, \epsilon, \epsilon) : c \\ \lambda re\phi.agent(r, z) \wedge \phi e \end{array}}\ [mg] \\ \lambda SOr.S(\lambda x.O(\lambda ye\phi.read(r, x, y) & \\ \wedge patient(r, y) \wedge \phi e)) & \end{array}}{\begin{array}{l} wh, k, d, k \otimes d \vdash (y\ z, lit, w\ x) : c \\ \lambda SOr.S(\lambda x.O(\lambda ye\phi.read(r, x, y) \wedge patient(r, y) \\ \wedge agent(r, x) \wedge \phi e)) \end{array}}\ [phase]$$

On constate l'ambiguïté sur l'ordre des transferts à réaliser, mais la normalisation nous assure que la preuve conduit au même résultat. Le minimalisme tend à favoriser la réalisation du déplacement en attente le plus récent, c'est-à-dire celui du sujet.

$$\dfrac{\begin{array}{l}\vdash (\epsilon, un, enfant) : (k \otimes d) \\ \lambda Qe\phi.\exists x.child(x) \wedge Qx(x :: e)\phi\end{array} \qquad \begin{array}{l}wh, k, d, k \otimes d \vdash (z\ y, lit, w\ x) : c \\ \lambda SOr.S(\lambda x.O(\lambda ye\phi.read(r,x,y) \\ \wedge patient(r,y) \wedge agent(r,x) \wedge \phi e))\end{array}}{\begin{array}{l}wh, k \otimes d \vdash (y\ un\ enfant, lit, x) : c \\ \lambda Or.\exists x.child(x) \wedge O(\lambda ye\phi.read(r,x,y) \\ \wedge patient(r,y) \wedge agent(r,x) \wedge \phi(x :: e)))\end{array}}\ [phase_t]$$ Par

ailleurs nous construisons le syntagme nominal interrogatif qui est utilisé dans la seconde opération du *transfert*.

$$\dfrac{\dfrac{\begin{array}{l}\vdash (\epsilon, Quel, \epsilon) : (wh \otimes (k \otimes d)) \,/\, n \\ \lambda PQe\phi?x.Px(x :: e)(\lambda e'.Qxe'\phi)\end{array} \qquad \begin{array}{l}\vdash (\epsilon, livre, \epsilon) : n \\ \lambda ze\phi.book(z) \wedge \phi e\end{array}}{\begin{array}{l}\vdash (\epsilon, Quel, livre) : (wh \otimes (k \otimes d)) \\ \lambda Qe\phi?x.book(x) \wedge Qx(x :: e)\phi\end{array}}\ [mg] \qquad \begin{array}{l}\vdots \\ wh, k \otimes d \vdash (y\ un\ enfant, lit, x) : c \\ \lambda Or.\exists x.child(x) \wedge O(\lambda ye\phi.read(r,x,y) \\ \wedge patient(r,y) \wedge agent(r,x) \wedge \phi(x :: e)))\end{array}}{\begin{array}{l}\vdash (Quel\ livre\ un\ enfant, lit, \epsilon) : c \\ \lambda re\phi.\exists x.child(x) \wedge ?y.book(y) \wedge read(r,x,y) \\ \wedge patient(r,y) \wedge agent(r,x) \wedge \phi(y :: x :: e)))\end{array}}\ [phase_t]$$

Dérivations non acceptées grâce à la PIC

Un autre apport de la modélisation des *phases* est la définition d'îlot. La *Phase Impenetrability Condition* de (Chomsky, 1999) est représentée par le fait que, dans la règle de *phase*, la partie en position de complément doit être vidée de ses hypothèses. Pour illustrer cet aspect de la modélisation, nous proposons un exemple fabriqué où l'hypothèse k utilisée par le syntagme nominal objet n'apparaît pas dans l'item lexical de mode. De fait, l'hypothèse d ne pourra pas être déchargée dans la dérivation, mais la *phase* permet de s'en rendre compte *immédiatement*.

L'entrée lexicale pour mode simulant ce problème serait alors $V; v \vdash (\epsilon, \epsilon, \epsilon) : d \setminus V$ qui produit une preuve ayant pour conclusion V, avec seulement une hypothèse d dans sa partie gauche $d, V; v \vdash (w\epsilon, \epsilon) : V$. La première partie de la phase produit :

$$\dfrac{d \vdash (\epsilon, lit, u) : (V \odot v) \quad d, V; v \vdash (w\ v, \epsilon, \epsilon) : V}{d, d \vdash (w\ v, lit, u) : V}\ [phase]$$

L'hypothèse k n'étant pas présente, le transfert ne peut être utilisé pour faire entrer l'objet. Dans la définition de la règle de la *phase*, la partie correspondant à Γ_c n'est pas vide et la dérivation s'arrête. Sur cet exemple simplifié, on voit l'encodage de la PIC dans la règle de la *phase* grâce aux propriétés des MCG. Il n'est alors pas besoin d'introduire de nouvelle règle *ad hoc* pour traiter une condition supplémentaire.

6. Conclusion

La proposition principale de cet article est l'introduction de la notion de phase dans des grammaires de type logique, les MCG. Pour cela une nouvelle règle est introduite dans le formalisme, ce qui permet d'utiliser la commutativité et la non-commutativité entre les éléments dans l'analyse. Cette nouvelle règle est la substitution d'hypothèses comparables (entretenant un ordre non-commutatif), suivi par un *transfert* qui réalise un ensemble de déplacements. Nous avons mis en avant le rôle particulier que jouent les *phases* au niveau syntaxique permettant de définir des îlots d'accessibilité avec la condition PIC qui est directement encodée par les propriétés logiques du formalisme.

Le calcul synchronisé produisant une représentation sous forme de chaînes de caractères n'est pas modifié, alors que le calcul sémantique l'est profondément. Grâce à l'utilisation des *phases* et de la modélisation des continuations dans le λ-calcul, ce calcul est réduit à l'application fonctionnelle. Les termes utilisés sont similaires à (de Groote, 2006), et la *phase* mobilise un terme permettant d'unifier les variables des différents termes. Pour la sémantique, les *phases* sont l'introduction des prédicats spécifiant les rôles thématiques, reliés aux variables par la réification des formules.

Le formalisme obtenu est motivé par la prise en compte de la capacité humaine dans la production et la compréhension langagière. Par ailleurs, d'un point de vue technique, l'utilisation de la logique permet d'obtenir un calcul sémantique sans augmenter la complexité. Enfin, ce formalisme est basé sur les MG qui génèrent des langages faiblement sensibles au contexte (*midly context sensitive languages*), et pour lesquelles il existent des conditions permettant de conserver une analyse polynomiale. Nous cherchons à conserver ces propriétés dans les MCG.

Une difficulté dans la mise en œuvre de cette proposition est la constitution de lexiques qui nécessite de bien connaître les propriétés du formalisme. Il est possible d'utiliser les lexiques proposés pour les GM au vue de l'isomorphisme entre les formalismes, mais des expériences de transformation de lexiques de grandes grammaires seraient des pistes intéressantes. Une autre voie est d'utiliser le cadre logique pour réaliser un apprentissage (inférence grammaticale), (Bonato et Retoré, 2014), à partir de corpus annotés comme (Moot, 2010), ou l'apprentissage λ-termes sémantique (Zettlemoyer et Collins, 2009). L'étude de l'équivalence entre MCG$_{phase}$ et MG doit être approfondie pour mieux définir la capacité générative du formalisme.

Remerciements

Je remercie les rapporteurs de la revue pour leurs précieux conseils qui ont permis une importante évolution de l'article, et les autres relecteurs pour leur patient travail.

7. Bibliographie

Ajdukiewicz K., « Die syntaktische Konnexität », *Studia Philosophica*, vol. 1, p. 1-27, 1935.

Amblard M., Calcul de représentations sémantiques et syntaxe générative : les grammaires minimalistes catégorielles, PhD thesis, université de Bordeaux 1, septembre, 2007.

Amblard M., « Minimalist Grammars and Minimalist Categorial Grammars, definitions toward inclusion of generated languages », *in* S. Pogodalla, M. Quatrini, C. Retoré (eds), *Logic and Grammar*, vol. 6700 of *LNCS/LNAI*, springer, p. 61-80, June, 2011.

Amblard M., Lecomte A., Retoré C., « Categorial Minimalist Grammar : From Generative Syntax To Logical Form », *Linguistic Analysis*, vol. 36, nº 1–4, p. 273-306, December, 2010.

Amblard M., Retoré C., « Partially Commutative Linear Logic and Lambek Caculus with Product : Natural Deduction, Normalisation, Subformula Property », *IFCoLog Journal of Logic and its Applications*, vol. 1, nº 1, p. 53-94, June, 2014. 41 pages.

Baker M., « Thematic Roles and Syntactic Structure », *in* L. Haegeman (ed.), *Elements of Grammar, Handbook of Generative Syntax*, Kluwer, Dordrecht, p. 73-137, 1997.

Bar-Hillel Y., « A quasi arithmetical notation for syntactic description », *Language*, vol. 29, p. 47-58, 1953.

Bonato R., Retoré C., « Learning Lambek grammars from proof frames », *in* C. Casadio, B. Coecke, M. Moortgat, P. Scott (eds), *Categories And Types In Logic, Language And Physics – Festschrift on the occasion of Jim Lambek's 90th birthday*, vol. 8222 of *LNCS/FoLLI*, Springer-Verlag, p. 108-135, 2014.

Bresnan J., *Lexical-Functional Syntax*, Blackwell Publishers, Oxford, UK, 2001.

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

Chomsky N., *The Minimalist Program*, MIT Press, Cambridge, 1995.

Chomsky N., *Derivation by phase*, ms, MIT, 1999.

de Groote P., « Partially commutative linear logic : sequent calculus and phase semantics », *in* V. M. Abrusci, C. Casadio (eds), *Third Roma Workshop : Proofs and Linguistics Categories – Applications of Logic to the analysis and implementation of Natural Language*, Bologna :CLUEB, p. 199-208, 1996.

de Groote P., « Towards Abstract Categorial Grammars », *ACL 2001*, 2001.

de Groote P., « Towards a Montagovian account of dynamics », *in* M. Gibson, J. Howell (eds), *Proceedings of Semantics and Linguistic Theory (SALT) 16*, 2006.

Guillaume B., Perrier G., « Interaction Grammars », *Research on Language and Computation*, vol. 7, nº 2-4, p. 171-208, 2009.

Hale, Keyser, « On Argument Structure and the Lexical Expression of Syntactic Relations », *The View from Building 20. Ithaca*, 1993.

Harkema H., « A Characterization of Minimalist Languages », *in* P. de Groote, G. Morrill, C. Retoré (eds), *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 193-211, 2001.

Joshi A., Schabes Y., « Tree-Adjoining Grammars », *in* G. Rozenberg, A. Salomaa (eds), *Handbook of Formal Languages*, Springer Berlin Heidelberg, p. 69-123, 1997.

Kaplan R. M., Bresnan J., « Lexical-Functional Grammar : A Formal System for Grammatical Representation », *in* J. Bresnan (ed.), *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, MA, p. 173-281, 1982.

Kobele G. M., Generating Copies : An investigation into structural identity in language and grammar, PhD thesis, UCLA, 2006.

Kratzer A., « External Arguments », *in* E. Benedicto, J. Runner (eds), *Functional Projections*, University of Massachussets, Occasional Papers, Amherst, 1994.

Lambek J., « The Mathematics of sentence structures », *American mathematical monthly*, 1958.

Lambek J., « Type Grammar Revisited », *in* A. Lecomte, F. Lamarche, G. Perrier (eds), *Logical Aspects of Computational Linguistics*, vol. 1582 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 1-27, 1999.

Lecomte A., « Categorial Grammar for Minimalism », *Language and Grammar : Studies in Mathematical Linguistics and Natural Language*, vol. CSLI Lecture Notes, n° 168, p. 163-188, 2005.

Lecomte A., « Semantics in Minimalist-Categorial Grammars », *Formal Grammar*, 2008.

Lecomte A., Retoré C., « Extending Lambek grammars : a logical account of minimalist grammars », *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, ACL 2001*, ACL, Toulouse, p. 354-361, July, 2001.

Michaelis J., « Derivational Minimalism Is Mildly Context–Sensitive », *in* M. Moortgat (ed.), *Logical Aspects of Computational Linguistics*, vol. 2014 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 179-198, 2001.

Michaelis J., « An Additional Observation on Strict Derivational Minimalism », *in* J. Rogers (ed.), *Proceedings of FG-MoL 2005. The 10th conference on Formal Grammar and The 9th Meeting on Mathematics of Language, pp. 101-111*, CSLI Publications, Stanford, 2009.

Moortgat M., « Categorial Type Logics », *in* J. van Benthem, A. ter Meulen (eds), *Handbook of Logic and Language*, Elsevier, chapter 2, p. 93-178, 1997.

Moot R., « Automated Extraction of Type-Logical Supertags from the Spoken Dutch Corpus », *in* S. Bangalore, A. Joshi (eds), *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing : A Supertagging Approach*, MIT Press, 2010.

Morrill G., « Type Logical Grammar », *Categorial Logic of Signs*, 1994.

Muskens R., « Language, Lambdas, and Logic », *in* G.-J. Kruijff, R. Oehrle (eds), *Resource-Sensitivity, Binding and Anaphora*, vol. 80 of *Studies in Linguistics and Philosophy*, Springer Netherlands, p. 23-54, 2003.

Pollard C., Convergent Grammars, Technical report, The Ohio State University., 2007.

Retoré C., « A description of the non-sequential execution of Petri nets in partially commutative linear logic », *Logic Colloquium 99*, vol. Lecture Notes in Logic, p. 152-181, 2004.

Ruet P., Fages F., « Concurrent constraint programming and non-commutative logic », *in* M. Nielsen, W. Thomas (eds), *Computer Science Logic*, vol. 1414 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 406-423, 1998.

Sag I. A., Pollard C. J., Head-Driven Phrase Structure Grammar : An Informal Synopsis, CSLI Report n° 87-79, Stanford University, Stanford University, 1987.

Stabler E., « Derivational Minimalism », *Logical Aspect of Computational Linguistic*, 1997.

Stabler E., « Recognizing Head Movement », *in* P. de Groote, G. Morrill, C. Retoré (eds), *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 245-260, 2001.

Steedman M., « Combinatory grammars and parasitic gaps », *Natural Language and Linguistic Theory 5*, 1987.

Zettlemoyer L. S., Collins M., « Learning Context-dependent Mappings from Sentences to Logical Form », *Volume 2*, ACL '09, Stroudsburg, PA, USA, p. 976-984, 2009.