Vers une implémentation de la théorie sens-texte avec les grammaires catégorielles abstraites

Marie Cousin

Vers une implémentation de la théorie sens-texte avec les grammaires catégorielles abstraites

Marie Cousin[1]
(1) Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
marie.cousin@loria.fr

RÉSUMÉ

La théorie sens-texte est une théorie linguistique visant à décrire la correspondance entre le sens et le texte d'un énoncé à l'aide d'un outil formel qui simule l'activité langagière d'un locuteur natif. Nous avons mis en place une premiere implémentation de cette théorie à l'aide des grammaires catégorielles abstraites, qui sont un formalisme grammatical basé sur le λ-calcul. Cette implémentation représente les trois niveaux de représentation sémantique, syntaxique profonde et syntaxique de surface de la théorie sens-texte. Elle montre que la transition de l'un à l'autre de ces niveaux (en particulier la génération d'une représentation de syntaxe de surface à partir d'une représentation sémantique d'un même énoncé) peut être implémentée en utilisant les propriétés avantageuses des grammaires catégorielles abstraites, dont la transduction.

ABSTRACT

Towards an implementation of meaning-text theory with abstract categorial grammars

Meaning-text theory is a linguistic theory aimed at describing the relation between meaning and text in natural language, with a formal device that simulates the linguistic activity of a native speaker. We established a first implementation of this theory with abstract categorial grammars, which are a grammatical formalism based on λ-calculus. This implementation represents the following three representation levels of meaning-text theory : semantic, deep syntactic and surface syntactic. It shows that transitions between these levels (especially the generation of a surface syntactic representation starting from a semantic representation of the same utterance) can be implemented using advantageous properties of abstract categorial grammars, like transduction.

MOTS-CLÉS : Grammaires Catégorielles Abstraites, Théorie Sens-Texte, Paraphrase.

KEYWORDS: Abstract Categorial Grammars, Meaning-Text Theory, Paraphrase.

1 Introduction

Cet article s'intéresse à l'implémentation de la théorie sens-texte (TST) à l'aide de grammaires catégorielles abstraites (ACG), afin de générer du texte. La TST est une théorie linguistique ayant déjà été utilisée en génération, tandis que les ACG sont un formalisme grammatical s'appuyant sur un système de types. Nous utiliserons donc des méthodes formelles. Ce choix est motivé par notre volonté d'avoir un texte très contrôlé, afin d'être sûrs que le texte généré est bien celui voulu et que le message transmis est bien le message que l'on voulait transmettre. Ces mêmes motivations se retrouvent aussi dans Grammatical Framework (Ranta, 2004), qui s'appuie, comme les ACG, sur un système de types. L'encodage et les liens vers d'autres formalismes dans les ACG a beaucoup été étudié (cf. tableau 1).

Étant donné que nous voulons avoir un très fort contrôle sur les différentes structures linguistiques aux différents niveaux, pour avoir les structures de la TST, nous avons privilégié une approche avec des méthodes formelles et non pas des méthodes numériques à base de réseaux de neurones.

Les ACG (de Groote, 2001) sont un formalisme, un cadre logique, utilisant des λ-termes et les propriétés du λ-calcul telles que la β-réduction. Ce cadre logique que sont les ACG permet de représenter d'autres formalismes grammaticaux. Les ACG ont de nombreux avantages, dont les suivants : elles sont réversibles, on peut donc les utiliser en génération ou analyse (Kanazawa, 2007). Leurs capacités à encoder d'autres formalismes, comme les TAG (Pogodalla, 2017a), et à être utilisées en génération ont été mises en oeuvre pour généraliser le modèle G-TAG (Danlos *et al.*, 2014). Elles sont également utilisées dans un cadre industriel chez Yseop. Le tableau ci-dessous (Pogodalla, 2017b) illustre le pouvoir expressif des ACG. Une hiérarchie des ACG est utilisée dans ce tableau, se basant sur deux notions (l'ordre et la complexité d'une ACG), qui sont définis plus bas en section 3.

ACG	langage généré
$\text{ACG}_{(1,n)}$	langages finis
$\text{ACG}_{(2,1)}$	langages réguliers
$\text{ACG}_{(2,2)}$	langages hors-contexte
$\text{ACG}_{(2,3)}$	grammaires hors-contexte multiples bien équilibrées
$\text{ACG}_{(2,4)}$	grammaires faiblement contextuelles
$\text{ACG}_{(2,4+n)}$	$\text{ACG}_{(2,4)}$
$\text{ACG}_{(3,n)}$	décidabilité de MELL

TABLE 1 – Pouvoir expressif des ACGs (Pogodalla, 2017b)

Dans cet article, nous cherchons à appliquer ces propriétés à une autre théorie linguistique ayant déjà démontré son intérêt pour les systèmes de génération : la TST.

La TST est en effet une théorie linguistique qui se donne pour objectif de décrire la correspondance entre le sens d'un énoncé et sa représentation sous forme textuelle par un outil formel, le modèle sens-texte, qui simule l'activité langagière d'un locuteur natif. Elle utilise entre autres une syntaxe de dépendances et les concepts clés de paraphrase et de fonctions lexicales (FL) (cf. sections suivantes), ces derniers permettant de rendre plus naturel le texte en question. En particulier, les FL syntagmatiques (cf. section 2) encodent les collocations ou encore les verbes supports. Elles sont donc très importantes, surtout lors des réalisations de surface. La direction privilégiée par cette théorie est la direction du sens vers le texte, soit la génération. Certaines formalisations et implémentations de ce modèle ont déjà eu lieu, telles que les grammaires polarisées et GUST (Kahane, 2005; Kahane & Lareau, 2005; Lareau, 2007) et MARQUIS (Wanner *et al.*, 2010; Lambrey & Lareau, 2015; Lareau *et al.*, 2018).

Nous cherchons ici à implémenter la TST avec les ACG et en particulier les structures linguistiques utilisées par la TST, même si pour chacun de ses niveaux de représentation il existe des formalismes relativement proches. Par exemple, au niveau sémantique la TST utilise en effet des graphes, les notions de prédicats et d'arguments, et se rapproche des AMR (Banarescu *et al.*, 2013). Au niveau de représentation syntaxique, elle utilise des arbres de dépendances, tout en ayant des types d'étiquettes différents d'autres formalismes en dépendance. Nous nous intéressons ici à l'articulation et l'unité au sein de la TST de ces structures linguistiques.

Une question se pose alors : les ACG sont-elles adaptées à implémenter une théorie linguistique basée sur une syntaxe de dépendance dans un but de génération ? Cet article présente la faisabilité d'une telle implémentation en nous appuyant sur un nombre restreint d'exemples qui mettent en œuvre plusieurs spécificités de la TST.

2 Théorie sens-texte et les fonctions lexicales

La TST (Mel'čuk *et al.*, 2012; Milićević, 2006) utilise le modèle sens-texte pour faire le lien entre le sens et le texte d'un énoncé. Le sens est le contenu linguistique à communiquer (Milićević, 2006) ; il n'est pas directement observable. Le texte est un fragment de discours (Milićević, 2006), et est immédiatement perceptible.

2.1 Théorie sens-texte

Le modèle sens-texte est composé de 7 niveaux de représentation : les niveaux de représentation sémantique (SemR), syntaxique profond (DSyntR), syntaxique de surface (SSyntR), morphologique profond (DMorphR), morphologique de surface (SMorphR), phonétique profond (DPhonR) et phonétique de surface (SPhonR) (cf. figure 1). Il comprend également 6 modules de transition entre ces niveaux, avec entre autres le module sémantique ↔ syntaxe profonde. Le modèle proposé par la TST permet donc d'analyser ou de générer un énoncé, selon le sens de transition choisi entre les niveaux.

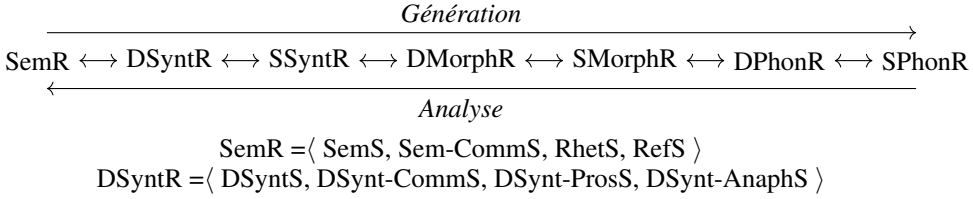


FIGURE 1 – Schéma du modèle sens-texte et détail des structures composant les représentations sémantiques et syntaxiques profondes (Mel'čuk *et al.*, 2012; Milićević, 2006)

Chacun de ces niveaux de représentation est composé de différentes structures : la structure principale ainsi que une ou plusieurs structures complémentaires (cf. figure 1 détaillant les structures dont sont composés les niveaux SemR et DSyntR). Par exemple, SemR est composé des structures sémantique (SemS, cf. figure 2 où cette structure est représentée pour deux expressions), sémantique communicative (Sem-CommS), sémantique rhétorique (RhetS) et sémantique de référence (RefS). La structure principale de SemR est SemS et est représentée sous forme de graphe orienté (cf. figure 2). Elle représente les liens (sémantiques) entre les sémantèmes (ou unités sémantiques) composant le sens exprimé. Mel'čuk *et al.* (2012) donnent plus de détail quant aux règles de bonne formation de cette structure. Sem-CommS indique (sur SemS, comme une sorte d'annotation) où vont se situer le thème et le rhème (cf. Milićević (2006), page 10, où un exemple est détaillé), RhetS va indiquer le style de l'expression (neutre, ironique, joyeux, etc.), et RefS est un ensemble de pointeurs partant des representations sémantiques vers les entités leur correspondant du monde réel. Cependant, ces trois dernières structures ne seront pas détaillées ou utilisées ici, seule l'est SemS, car c'est sur

celle-ci que se concentre cet article et notre implémentation. Les structures principales de DSyntR et SSyntR, respectivement DSyntS et SSyntS, sont représentées sous la forme d'arbres de dépendance (cf. figure 5). Ici aussi, pour les mêmes raisons que précédemment, nous ne détaillerons pas les autres structures de DSyntR et SSyntR. Les structures principales des autres niveaux de représentation de la TST sont représentées par des chaînes de caractères.

Dans ces modules de transition, ainsi qu'au sein de certains niveaux de représentation, la TST utilise les concepts clés de paraphrase et de fonctions lexicales (FL).

2.2 Fonctions lexicales

Les FL ont pour but de décrire les phénomènes linguistiques existant au sein des langages. Ce sont en effet des fonctions qui décrivent les relations entre les unités lexicales. Une FL va associer à une unité lexicale un ensemble constitué de toutes les autres unités lexicales alternatives correspondant à la relation qu'elle décrit (voir ci-dessous où des exemples sont donnés). Elles sont utilisées dans la TST dans le dictionnaire explicatif combinatoire (nous ne décrirons pas cet outil, voir Mel'čuk *et al.* (2012, 2013) pour plus de détails) ainsi que pour effectuer l'une des étapes de paraphrase ayant lieu lors de la génération.

Il existe deux grands types de FL : les FL paradigmatiques et les FL syntagmatiques (Mel'Čuk & Polguère, 2021). Les FL paradigmatiques sont relatives aux propriétés sémantiques des unités lexicales, alors que les FL syntagmatiques expriment les propriétés combinatoires des unités lexicales. Par exemple, `anti` est une FL paradigmatique qui à un lexème associe son contraire : `anti`(CALM) = {UPSET, RESTLESS} (à partir de Mel'čuk *et al.* (2013)), et `causFunc` une FL syntagmatique qui à un lexème L associe un verbe support signifiant *faire en sorte que* L *existe* : `causFunc`(ATTENTION) = DRAW (pour l'expression « *to draw attention* ») (Milićević, 2006). Les FL permettent ainsi d'encoder des phénomènes lexicaux tels que les collocations ou les verbes supports. Plus de détails sur les FL sont donnés dans Mel'Čuk & Polguère (2021). Nous ne détaillerons pas plus cette notion ici.

2.3 Paraphrase

Lors de la génération d'un énoncé à partir d'une représentation sémantique, la TST utilise plusieurs étapes de paraphrase (Iordanskaja *et al.*, 1991) : la paraphrase sémantique (entre SemR et DSyntR), la paraphrase syntaxique profonde (au niveau de DSyntR), la réalisation des FL (entre DSyntR et SSyntR), et au moment de choisir la réalisation de syntaxe de surface (au niveau de SSyntR). Chacune de ces étapes de paraphrase est effectuée par un module de transitions. Nous nous intéressons dans cet article à la paraphrase sémantique.

La figure 2 illustre un exemple de paraphrase au niveau de représentation sémantique de la TST. La paraphrase sémantique est basée sur la composition des sens. En effet, chaque sémantème est défini dans la TST par des sémantèmes plus simples. Ainsi, les sens sont "dépliables" (en une sorte de définition). Un sens est dit composé s'il peut être défini par des sens plus simples. Sinon, il est dit primitif (Mel'čuk *et al.*, 2012). La figure 2 illustre ce phénomène en "dépliant" ⟨assassinate⟩.

Nous nous plaçons ici dans un contexte de génération. Nous nous occuperons principalement de SemR et DSyntR, et nous donnerons en section 5 un exemple détaillé.

(a) « *Charlie assassinates Taylor.* » (b) « *Charlie murders Taylor intentionally.* »

FIGURE 2 – Représentation de deux structures sémantiques illustrant l'équivalence de deux graphes pour la paraphrase sémantique de « *Charlie assassinates Taylor* » (à partir de Mel'čuk *et al.* (2013)).

3 Grammaires catégorielles abstraites (ACG)

Les ACG (de Groote (2001), dont nous reprenons ici les définitions) sont un formalisme grammatical basé sur le λ-calcul. Une ACG est composée de deux langages, liés par un lexique. Le premier langage est appelé *langage abstrait* et correspond à l'ensemble des structures grammaticales abstraites, comme des arbres d'analyse. Le second langage, le *langage objet*, correspond à l'ensemble des réalisations concrètes générées par le langage abstrait, tel que des chaînes de caractère ou des représentations logiques sous forme de graphe. Chacun de ces deux langages est un ensemble de λ-termes obtenu par induction sur une signature.

Définition 1 *Soit A l'ensemble des types atomiques. On note $\mathcal{T}(A)$ l'ensemble des* ***types implicatifs linéaires*** *obtenu par induction sur A :*

— *si $a \in A$ alors $a \in \mathcal{T}(A)$*
— *si $\alpha, \beta \in \mathcal{T}(A)$ alors $(\alpha \to \beta) \in \mathcal{T}(A)$*

Définition 2 *Soit Σ une* ***signature d'ordre supérieur****. Σ est de la forme $\Sigma = \langle A, C, \tau \rangle$, où :*

— *A est un ensemble de types atomiques,*
— *C un ensemble de constantes,*
— *$\tau : C \longrightarrow \mathcal{T}(A)$ une fonction.*

Pour indiquer qu'un λ-terme t est de type s dans la signature Σ, on note $\vdash_\Sigma t : s$, ou encore $t : s$ s'il n'y a pas d'ambiguïté.

On note $\Lambda(\Sigma)$ l'ensemble des λ-termes obtenus en utilisant les constantes de C, les variables, les abstractions et les applications.

Définition 3 *Étant donné deux signatures Σ_1 et Σ_2, un* ***lexique*** *$\mathcal{L}_{12}$ de Σ_1 dans Σ_2 est un couple de morphismes $\langle F, G \rangle$ tel que $F : \tau(A_1) \longrightarrow \tau(A_2)$ et $G : \Lambda(\Sigma_1) \longrightarrow \Lambda(\Sigma_2)$.*

On note alors $\mathcal{L}_{12}(t) = \gamma$ l'interprétation de t par $\mathcal{L}_{12}$ (qui est alors égale à γ), ou encore $t := \gamma$ s'il n'y a pas d'ambiguïté sur le lexique utilisé.

Les signatures et ACG détaillées ici utilisent des λ-termes presque linéaires. Nous ne détaillerons cependant pas cette notion car elle n'a que peu d'intérêt par rapport à ce que nous exposons ici, les variables utilisées n'étant ni effacées ni dupliquées dans les lexiques. Nous notons toutefois λ^o

et λ pour les abstractions respectivement linéaires et non linéaires, et $\rightarrow$ et $\Rightarrow$ pour les types non atomiques respectivement linéaires et non linéaires.

Nous définissons ainsi $\Sigma_{\textit{semantic}}$ qui correspond au niveau SemR (et permet de représenter les SemS illustrées en figure 2), et qui illustre la définition 2, de la manière suivante :

— $A_{semantic} = \{n, t\}$
— $C_{semantic} = \{\textit{true},\ c^s_{intention},\ c^s_{Charlie},\ c^s_{Taylor},\ c^s_{murder},\ R_1,\ R_2\}$
— $\tau_{semantic}$ est donné par la table 2

Constante		Type
$c^s_{intention}$	:	$n \Rightarrow t$
$c^s_{Charlie}$	:	$n \Rightarrow t$
c^s_{Taylor}	:	$n \Rightarrow t$
c^s_{murder}	:	$n \Rightarrow t$
R_1	:	$n \rightarrow t \rightarrow t$
R_2	:	$n \rightarrow t \rightarrow t$
$true$	:	t

TABLE 2 – $\tau_{semantic}$

Pour des raisons de concision, nous ne définirons plus par la suite les ensembles A et C ; ils s'induisent de la table représentant τ. Nous pouvons maintenant définir formellement une ACG et les notions de langages abstrait et objet :

Définition 4 *Une **grammaire catégorielle abstraite** est un quadruplet $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s\rangle$ où :*
— *$\Sigma_1 = \langle A_1, C_1, \tau_1\rangle$ et $\Sigma_2 = \langle A_2, C_2, \tau_2\rangle$ sont deux signatures d'ordre supérieur,*
— *$\mathcal{L} = \Sigma_1 \longrightarrow \Sigma_2$ est le lexique,*
— *$s \in \mathcal{T}(A_1)$ est le type distingué de la grammaire.*

Définition 5 *Le **langage abstrait** $\mathcal{A}$ et le **langage objet** $\mathcal{O}$ d'une ACG $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s\rangle$ sont :*
— *$\mathcal{A} = \{t \in \Lambda(\Sigma_1) |\ \vdash_{\Sigma_1} t : s$ est dérivable$\}$*
— *$\mathcal{O} = \{t \in \Lambda(\Sigma_2) |\ \exists\, u \in \mathcal{A}(\mathcal{G})$ tel que $t = \mathcal{L}(u)\}$*
Nous utilisons dans cet article la $\beta\eta$-equivalence comme égalité entre les λ-termes

Afin d'illustrer ces concepts, nous introduisons la signature abstraite $\Sigma_{\textit{deep-syntactic}}$ (représentée table 3). Nous pouvons maintenant définir le lexique $\mathcal{L}_{sem}$ (représenté table 4), qui formera avec les signatures $\Sigma_{\textit{semantic}}$ et $\Sigma_{\textit{deep-syntactic}}$ une ACG. Cette ACG permettra la première partie de la transition de SemR ($\Sigma_{\textit{semantic}}$) vers DSyntR ($\Sigma_{\textit{dsynt-tree}}$) puisque $\Sigma_{\textit{deep-syntactic}}$ correspond à une représentation syntaxique profonde abstraite.

Constante		Type
$c^{ds}_{assassinate}$	:	$G' \rightarrow G \rightarrow G$
$c^{ds}_{intentionally}$	:	$(MOD \rightarrow G' \rightarrow G) \rightarrow G' \rightarrow G$
$c^{ds}_{Charlie}$	:	G
c^{ds}_{Taylor}	:	G
c^{ds}_{murder}	:	$MOD \rightarrow G' \rightarrow G \rightarrow G$
EXP	:	$G \rightarrow G'$

TABLE 3 – $\tau_{deep-syntactic}$

Nous définissons également les notions d'ordre et de complexité d'une ACG. Ces notions sont utilisées dans la table 1 qui décrit le pouvoir expressif des ACG.

$\Sigma_{deep\text{-}syntactic}$		$\Sigma_{semantic}$
G	:=	$n \Rightarrow t$
G'	:=	$n \Rightarrow t$
MOD	:=	t
$c^{ds}_{assassinate}$	:=	$\lambda^{o}\ x\ y.\ \lambda\ e_0.\ (\exists e_x.\ \exists\ e_y.\ (c^{s}_{murder}\ e_0) \wedge (R_1\ e_0\ e_x) \wedge (x\ e_x) \wedge (y\ e_y)$ $\wedge(\exists e_1.\ (c^{s}_{intention}\ e_1) \wedge (R_1\ e_1\ e_x) \wedge (R_2\ e_1\ e_0)))$
$c^{ds}_{intentionally}$	:=	$\lambda^{o}\ pred.\ \lambda^{o}\ x.\ \lambda\ e0.\ \exists\ e_1.\ (c^{s}_{intention}\ e_1) \wedge (R_2\ e_1\ e_0)$ $\wedge(pred\ true\ (\lambda\ e_2.\ (R_1\ e_1\ e_2) \wedge (e_x\ e_2))\ e_0)$
$c^{ds}_{Charlie}$	:=	$\lambda\ e_0.\ (c^{s}_{Charlie}\ e_0)$
c^{ds}_{Taylor}	:=	$\lambda\ e_0.\ (c^{s}_{Taylor}\ e_0)$
c^{ds}_{murder}	:=	$\lambda^{o}\ A.\ \lambda^{o}\ x\ y.\ \lambda\ e_0.\ (\exists\ e_x.\ \exists\ e_y.\ (c^{s}_{murder}\ e0) \wedge (R_1\ e_0\ e_x)$ $\wedge(R_2\ e_0\ e_y) \wedge A \wedge (x\ e_x) \wedge (y\ e_y))$

TABLE 4 – $\mathcal{L}_{sem}$

Définition 6 (Pogodalla, 2017b) *L'**ordre** d'une ACG est le maximum de l'ordre de ses constantes abstraites. L'ordre d'une constante abstraite est l'ordre de son type τ. L'ordre d'un type $\tau \in \mathcal{T}(A)$ est défini par induction :*

— $ordre(\tau) = 1$ *si* $\tau \in A$,

— $ordre(\alpha \to \beta) = max(1 + ordre(\alpha), ordre(\beta))$ *sinon.*

*La **complexité** d'une ACG est le maximum des ordres des réalisations de ses types atomiques. Une ACG d'ordre γ et de complexité η est notée $ACG_{(\gamma,\eta)}$.*

Nous notons dans cet article c^{X}_{L} pour la constante de Σ_X représentant le lexème L . Nous utilisons aussi la notation γ^{X}_{i} pour le λ-terme complexe de $\Lambda(\Sigma_X)$ indexé par i. Ces λ-termes complexes encodent en fait une représentation (dans $\Lambda(\Sigma_X)$) d'une expression, expression que l'on indique grâce à i. Ainsi, nous utiliserons l'indice a pour « *Charlie assassinates Taylor* » et l'indice m pour « *Charlie murders Taylor intentionally* ». Nous notons respectivement s, ds et dt au lieu de $semantic$, $deep - syntactic$ et $dsynt - tree$ dans cette notation.

On définit alors les λ-termes complexes grâce aux tables 2, 3 et 4 :

$$\gamma^{s}_{m} = \lambda^{o}\ A.\ \lambda^{o}\ x\ y.\ \lambda\ e_0.\ (\exists\ e_x.\ \exists\ e_y.\ (c^{s}_{murder}\ e0) \wedge (R_1\ e_0\ e_x) \wedge (R_2\ e_0\ e_y) \wedge A \wedge (c^{s}_{Charlie}\ e_x) \wedge (c^{s}_{Taylor}\ e_y)) \quad (1)$$

$$\gamma^{ds}_{m} = (c^{ds}_{intentionally}(\lambda^{o}\ A\ x.\ c^{ds}_{murder}\ A\ x\ c^{ds}_{Taylor}))(EXP\ c^{ds}_{Charlie}) \quad (2)$$

$$\gamma^{ds}_{a} = c^{ds}_{assassinate}\ (EXP\ c^{ds}_{Charlie})\ c^{ds}_{Taylor} \quad (3)$$

Une autre propriété des ACG est la transduction : étant donné deux ACG partageant la même signature abstraite, la transduction (cf. figure 3) consiste en la composition de l'analyse (c'est-à-dire une inversion de morphisme, comme ${\mathcal{L}_{sem}}^{-1}$ dans la figure 3) et de l'application (c'est-à-dire une application de morphisme, comme $\mathcal{L}_{dsyntRel}$ dans la figure 3) à l'aide de chacune des ACG. La transduction permet donc de mettre en relation les termes de deux languages objets.

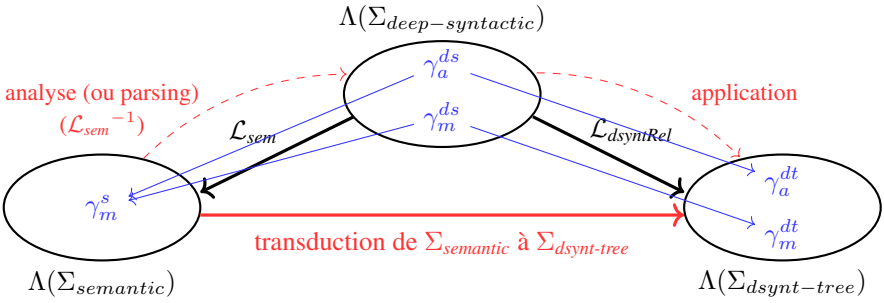


FIGURE 3 – Illustration de la transduction dans le cas de la transition de SemR à DSyntR par inversion de $\mathcal{L}_{sem}$ et sa composition avec $\mathcal{L}_{dsyntRel}$. Le lexique $\mathcal{L}_{sem}$ est tel que $\mathcal{L}_{sem}(\gamma_a^{ds}) = \mathcal{L}_{sem}(\gamma_m^{ds})$

4 Implémentation et résultats obtenus

Nous avons implémenté différentes signatures et lexiques afin de modéliser le fonctionnement du modèle sens-texte de SemR à SSyntR. Cependant, pour des soucis de simplification et cette implémentation se voulant exploratoire, nous n'avons représenté que les principaux niveaux de représentation de la TST, et non pas les structures communicatives ou rhétoriques par exemple (cf. figure 1).

Cette implémentation exploite la transduction (cf. figure 3), c'est-à-dire la composition entre une inversion de morphismes et l'application de morphismes dans les ACG. C'est en effet un moyen de calcul qui va permettre la transition entre les modules (ou les signatures objet dans notre implémentation) pour la transformation des structures. Dans notre implémentation, cette propriété de transduction est cruciale et exploitée afin de réaliser toutes les transformations. En effet, étant donné un λ-terme initial, on obtient par transduction un second λ-terme, sans avoir modifié le premier terme. Nous allons donc utiliser la transduction, qui nous semble particulièrement judicieuse, car la TST est comme une cascade de transformations de structures si l'on regarde comment elle est construite (cf. figure 1). Cette cascade se devine également sur la figure 4 (qui représente l'architecture des ACG dans notre implémentation) dans les zones 1, 2, 4 et 5.

Cependant, la zone 3 (cf. figure 4) forme comme une excroissance. En effet, lors de la génération, bien que la TST ne modifie pas les structures de représentation d'un niveau à l'autre (similairement à la transduction), nous voulons garder les premières structures. Dans les étapes de paraphrase par exemple, il se peut que plusieurs paraphrases différentes soient possibles (cf. table 5 illustrant le nombre de structures syntaxiques profondes obtenues après la paraphrase syntaxique profonde). Nous voulons donc garder une trace de toutes les applications de règles de paraphrases (initiales, intermédiaires et finales), et ce en particulier lors de l'étape de paraphrase syntaxique profonde, ce qui n'est pas propre à la transduction. Autrement dit, la TST fait de la réécriture tout en restant dans un même niveau de représentation, et sans supprimer les structures antérieures du même niveau : elles restent toutes dans l'ensemble qui nous intéresse, et doivent toutes passer à l'étape suivante. Cette étape est particulièrement compliquée à implémenter avec les ACG, car cela forme une boucle. La zone 3 représente cette boucle de paraphrase syntaxique profonde. Il faut, pour chacune des structures syntaxiques profondes, effectuer la transduction de $\Sigma_{dsynt\text{-}tree}$ à $\Sigma_{dsynt\text{-}rule}$, puis de $\Sigma_{dsynt\text{-}rule}$ à $\Sigma_{dsynt\text{-}tree}$, et recommencer jusqu'à ce qu'aucune nouvelle structure ne soit renvoyée. Nous verrons par la suite

que cette boucle est problématique.

La figure 4 illustre les liens entre les différentes ACG implémentées. On retrouve les différents niveaux de représentation de la TST : $\Sigma_{semantic}$ correspond au niveau SemR, $\Sigma_{dsynt\text{-}tree}$ correspond au niveau DSyntR, $\Sigma_{dsynt\text{-}0\text{-}fl}$ est une étape intermédiaire correspondant au niveau DSyntR mais où les FL auront été réalisées, et $\Sigma_{ssynt\text{-}tree}$ correspond au niveau SSyntR. La transduction (cf. figure 3) est exploitée ici, et a lieu :

— entre les signatures $\Sigma_{semantic}$ et $\Sigma_{dsynt\text{-}tree}$: pour faire la transition du niveau sémantique à celui de syntaxe profonde (il s'agit des zones 1 et 2, qui seront détaillées en section 5),
— entre les signatures $\Sigma_{dsynt\text{-}tree}$ et $\Sigma_{dsynt\text{-}rule}$: pour effectuer la paraphrase syntaxique profonde (zone 3),
— entre les signatures $\Sigma_{dsynt\text{-}tree}$ et $\Sigma_{dsynt\text{-}0\text{-}fl}$: pour réaliser les FL (zone 4),
— entre les signatures $\Sigma_{dsynt\text{-}0\text{-}fl}$ et $\Sigma_{ssynt\text{-}tree}$: pour faire la transition entre la syntaxe profonde et la syntaxe de surface (zone 5).

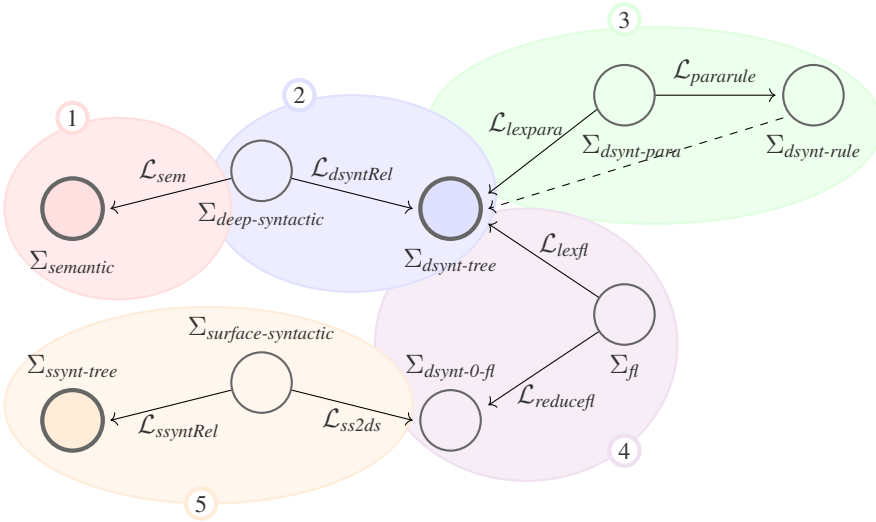


FIGURE 4 – Aperçu de l'architecture des ACG. La zone 1 correspond à l'étape de paraphrase sémantique, la zone 2 à la transition entre la sémantique et la syntaxe profonde, la zone 3 à l'étape de paraphrase syntaxique profonde, la zone 4 à l'étape de réalisation des FL et la zone 5 à la transition entre une représentation de syntaxe profonde où les FL seraient réalisées et la syntaxe de surface.

Cette implémentation a été réalisée avec le logiciel ACGtk (Pogodalla, 2016), et testée sur un ensemble de phrases exemples. Cet ensemble d'exemples est certes restreint, mais couvre plusieurs phénomènes linguistiques, tels que les collocations, le calcul et la manipulation des FL, les équivalences sémantiques ou syntaxiques entre deux énoncés, ou les arguments optionnellement exprimables par exemple. La table 5 illustre le nombre de structures obtenues par étape de génération pour deux graphes sémantiques pris comme exemples, correspondant aux expressions « *Charlie murders Taylor intentionnally* » et « *Alain is calm* ». La première, détaillée dans cet article, illustre la paraphrase sémantique et le traitement des groupes adverbiaux. La seconde, non détaillée ici par manque de place, illustre la paraphrase syntaxique profonde ainsi que le traitement des FL. En outre, les structures d'un niveau de représentation n'étant pas destinées à être réalisées dans le niveau de représentation suivant

dans la TST (parce qu'elles sont incorrectes par exemple) ne posséderont aucun antécédent dans la signature correspondant à ce niveau suivant (cf. table 5 montrant ce phénomène). Si une structure ne doit mener à rien par rapport au formalisme de la TST, alors notre implémentation des ACG est telle que, par transduction, aucun antécédent ne sera trouvé.

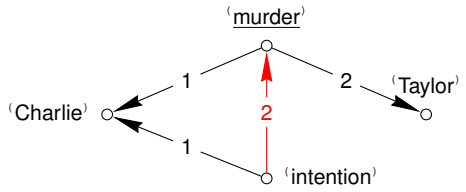
Le code complet contenant les ACG et les exemples est disponible à cette adresse. En effet, les exemples d'ACG que vous pouvez trouver dans cet article ont été simplifiés pour des soucis de clarté et de place.

Expression associée au graphe sémantique	« *Charlie murders Taylor intentionally* »	« *Alain is calm* »
SemR ($\Sigma_{semantic}$)	1	1
DSyntR avant paraphrase syntaxique profonde ($\Sigma_{dsynt\text{-}tree}$)	2	1
dont seront acceptées à l'étape suivante	2	1
DSyntR après paraphrase syntaxique profonde ($\Sigma_{dsynt\text{-}tree}$)	2	6 [1]
dont seront acceptées à l'étape suivante	2	3
DSyntR après réalisation des FL ($\Sigma_{dsynt\text{-}0\text{-}fl}$)	2	5
dont seront acceptées à l'étape suivante	2	3
SSyntR ($\Sigma_{ssynt\text{-}tree}$)	2	3

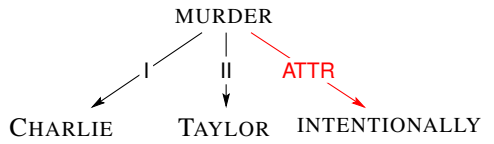
TABLE 5 – Nombre de structures obtenues par étape de la génération pour deux graphes sémantiques initiaux, correspondant aux expressions « *Charlie murders Taylor intentionally* » et « *Alain is calm* ».

De plus, cette implémentation permet, outre les transitions d'un niveau à un autre, le traitement des groupes adverbiaux, qui ont un fonctionnement particulier inspiré du travail sur les TAG de Pogodalla 2017a. Leur traitement n'est en effet pas le même selon que l'on se trouve dans $\Lambda(\Sigma_{semantic})$ ou $\Lambda(\Sigma_{dsynt\text{-}tree})$. Si l'on compare la direction de l'arc reliant le groupe adverbial au reste du graphe sémantique dans SemR et celle de la branche reliant le groupe adverbial au reste de l'arbre de syntaxe profonde dans DSyntR (cf. figure 5), alors on remarque qu'elles sont inversées. En effet, le groupe adverbial est un modificateur, son comportement change alors du comportement "standard" des autres unités lexicales : il pointe sur le sémantème qu'il modifie dans SemS, et est pointé par le lexème qu'il modifie en DSyntS (Mel'čuk *et al.*, 2013, 2015). Nous avons utilisé la même approche que les TAG pour les arbres de dérivation et les arbres dérivés lors des inversions de dépendances (Candito & Kahane, 1998). Notre implémentation permet aussi de ne pas exprimer un argument obligatoire mais optionnellement exprimable d'une SemR (Mel'čuk *et al.*, 2015). Ce dernier point est effectué à l'aide de constantes (EXP, IMP et I_MOD) de $\Sigma_{deep\text{-}syntactic}$, et a été inspiré de Blom *et al.* (2011) mais ne sera pas détaillé ici. Cependant, le traitement des arguments optionnellement exprimables est détaillé dans Cousin (2022), et la section 5 détaille l'exemple de paraphrase sémantique de la figure 2. Cet exemple illustre bien les différentes équivalences pouvant avoir lieu entre deux ACG en implémentant l'équivalence sémantique vue en figure 3, ainsi que le traitement des groupes adverbiaux avec « *intentionally* ».

1. En réalité, plus de structures sont obtenues, mais elles sont incorrectes par construction, elles ne sont donc pas considérées ici (et n'ont dans tous les cas pas d'antécédent dans $\Sigma_{surface\text{-}syntactic}$).



(a) Illustration de la SemS de « *Charlie murders Taylor intentionally.* », encodée par γ_m^s



(b) Illustration de la DSyntS de « *Charlie murders Taylor intentionally* », encodée par γ_m^{dt}

FIGURE 5 – Illustration de l'inversion de dépendance (en rouge) pour le cas de l'adverbe « *intentionally* » dans « *Charlie murders Taylor intentionally* »

5 Un exemple détaillé

Nous donnons dans cette section un exemple détaillé de transduction entre deux signatures et des équivalences pouvant intervenir. Nous nous plaçons dans le cadre de la transition de SemR à DSyntR, soit entre les signatures $\Sigma_{semantic}$ et $\Sigma_{dsynt\text{-}tree}$ (cf. figure 3). Nous n'allons nous intéresser qu'à l'équivalence sémantique dans cet exemple, c'est-à-dire deux termes ayant une DSyntR différente mais qui sont associées au même graphe sémantique, soit à la même SemR.

Prenons l'exemple de la paraphrase sémantique entre les deux expressions suivantes (inspiré de l'exemple de (Mel'čuk *et al.*, 2013), page 207) dont les graphes sémantiques donnés par la TST sont illustrés en figure 2, et dont la paraphrase sémantique est donnée en figure 3 :

(4) a. « *Charlie assassinates Taylor* »

b. « *Charlie murders Taylor intentionally* »

Nous voulons, en notant $\mathcal{T}$ la relation de transduction, avoir les équations suivantes :

$$\mathcal{L}_{sem}(\gamma_m^{ds}) = \gamma_m^s = \mathcal{L}_{sem}(\gamma_a^{ds}) \tag{5}$$

$$\mathcal{L}_{dsyntRel}(\gamma_m^{ds}) = \gamma_m^{dt} \tag{6}$$

$$\mathcal{L}_{dsyntRel}(\gamma_a^{ds}) = \gamma_a^{dt} \tag{7}$$

$$\mathcal{T}(\gamma_m^s,\ \gamma_m^{dt}) \text{ et } \mathcal{T}(\gamma_m^s,\ \gamma_a^{dt}) \tag{8}$$

$$\gamma_m^{dt} \equiv \gamma_a^{dt} \tag{9}$$

Les tables 2, 3, 4, 6 et 7 indiquent les termes des signatures et lexiques que nous allons utiliser dans cette section. Ces trois signatures et deux lexiques permettent la transition de SemR à DSyntR.

Nous avons choisi de modéliser la paraphrase sémantique en exploitant la transduction et les propriétés de β-réduction (entre autres) du λ-calcul : en effet, ces deux expressions ((4a) et (4b)) vont partager la même représentation sémantique (cf. section 2 et figure 2) au niveau de $\Sigma_{semantic}$ mais auront deux arbres de syntaxe profonde différents au niveau de $\Sigma_{dsynt\text{-}tree}$. Dans la modélisation de ce lien de paraphrase, nous utilisions des équivalences sur plusieurs niveaux différents. Cet exemple est ainsi bien adapté pour illustrer les différents niveaux d'équivalence sur lesquels nous pouvons travailler.

Constante		Type
lex_0	:	$l \to T$
lex_2	:	$l \to rel \to T \to rel \to T \to T$
lex_3	:	$l \to rel \to T \to rel \to T \to rel \to T \to T$
A_1	:	rel
A_2	:	rel
$ATTR$	:	rel
$c^{dt}_{assassinate}$	:	l
$c^{dt}_{intentionally}$	:	l
$c^{dt}_{Charlie}$	:	l
c^{dt}_{Taylor}	:	l
c^{dt}_{murder}	:	l

TABLE 6 – $\tau_{dsynt-tree}$

$\Sigma_{deep\text{-}syntactic}$		$\Sigma_{dsynt\text{-}tree}$
G	:=	T
G'	:=	T
MOD	:=	T
EXP	:=	λ^{o} LEX. LEX
$c^{ds}_{assassinate}$	:=	λ^{o} X Y. $lex_2\ c^{dt}_{assassinate}\ A_1$ X A_2 Y
$c^{ds}_{intentionally}$	:=	λ^{o} LEX. λ^{o} X. LEX $(lex_0\ c^{dt}_{intentionally})$X
$c^{ds}_{Charlie}$	:=	$lex_0\ c^{dt}_{Charlie}$
c^{ds}_{Taylor}	:=	$lex_0\ c^{dt}_{Taylor}$
c^{ds}_{murder}	:=	λ^{o} A. λ^{o} X Y. $lex_3\ c^{dt}_{murder}\ A_1$ X A_2 Y $ATTR$ A

TABLE 7 – $\mathcal{L}_{dsyntRel}$

Les termes γ^{dt}_a et γ^{dt}_m de la figure 3 sont construits grâce aux constantes de $\Sigma_{dsynt\text{-}tree}$ (illustrée table 6), et sont respectivement égaux à :

$$\gamma^{dt}_a = lex_2\ c^{dt}_{assassinate}\ A_1\ (lex_0\ c^{dt}_{Charlie})\ A_2\ (lex_0\ c^{dt}_{Taylor}) \tag{10}$$

$$\gamma^{dt}_m = lex_3\ c^{dt}_{murder}\ A_1\ (lex_0\ c^{dt}_{Charlie})\ A_2\ (lex_0\ c^{dt}_{Taylor})\ ATTR\ (lex_0\ c^{dt}_{intentionally}) \tag{11}$$

Ils correspondent aux DSyntR des expressions (4a) et (4b) respectivement.

Ces niveaux d'équivalence en question sont les suivants (cf. figure 3) :

— au sein de $\Sigma_{semantic}$ (cf. table 2) et au niveau du parsing par $\mathcal{L}_{sem}$ (cf. table 4) : les deux phrases (4a) et (4b) ont deux représentations différentes (γ^{ds}_m et γ^{ds}_a) dans $\Sigma_{deep\text{-}syntactic}$ (cf. table 3) mais le même graphe sémantique, qui est lui représenté au niveau de $\Sigma_{semantic}$ (par γ^{s}_m). On obtient donc, par β-réduction, l'égalité (12) modulo β. En effet, nous utilisons la β-équivalence au niveau d'une signature (ici $\Sigma_{semantic}$), et nous utilisons cette équivalence lorsqu'on dit que deux termes ont la même interprétation, car nous raisonnons en utilisant cette β-équivalence. (Les deux termes $\mathcal{L}_{sem}(\gamma^{ds}_m)$ et $\mathcal{L}_{sem}(\gamma^{ds}_a)$ sont bien équivalents à l'ordre des sous-expressions près : la représentation des termes n'est en effet pas unique. C'est d'ailleurs un problème sur lequel nous devons encore travailler.)

$$\mathcal{L}_{sem}(\gamma^{ds}_m) =_{\beta} \mathcal{L}_{sem}(\gamma^{ds}_a) \tag{12}$$

— au niveau de la transduction : les deux arbres de DSyntR et leurs représentations γ_a^{dt} et γ_m^{dt} dans $\Sigma_{dsynt\text{-}tree}$ (cf. table 6) sont equivalents. En effet (cf. figure 3), par application des lexiques $\mathcal{L}_{sem}$ (cf. table 4) et $\mathcal{L}_{dsyntRel}$ (cf. table 7), on a bien les égalités (5), (6) et (7) ci-dessus, puis (8), et donc (9) par transduction.

La transduction entre les signatures $\Sigma_{semantic}$ et $\Sigma_{dsynt\text{-}tree}$ permet donc de modéliser la paraphrase ayant lieu au niveau sémantique.

6 Conclusion

Nous avons montré ici une implémentation possible de la TST avec des ACG. Cette implémentation a eu lieu entre les niveaux SemR et SSyntR du modèle sens-texte. Bien que n'utilisant uniquement les structures principales des trois niveaux de représentation en question, et pas leurs autres structures, comme la structure communicative notamment, cette implémentation a montré, sur un ensemble de phrases exemples, que leurs structures syntaxiques de surface pouvaient être obtenues correctement (cf. table 5). En effet, pour un niveau de représentation donné, les structures incorrectes ne sont pas produites, car elles ne trouvent aucun antécédent par parsing du lexique menant à la signature abstraite associée à ce lexique. Il y a ainsi des moments de la génération où l'on produit des structures et d'autres où l'on filtre les structures obtenues. En outre, si l'on teste la capacité d'analyse de ce modèle, nous obtenons également les graphes sémantiques voulus.

Le modèle réalisé permet d'effectuer la paraphrase sémantique, les transitions entre les niveaux de représentation grâce à la transduction, ainsi que la réalisation des FL, également grâce à la transduction des ACG. De plus, le traitement des arguments optionnellement exprimables et des groupes adverbiaux est aussi permis.

Cependant, ce modèle montre quelques limites. En effet, la paraphrase syntaxique profonde n'est pas optimale. Elle est actuellement possible, et réalisée par l'excroissance dont nous parlions en section 4 (cf. figure 4, zone 3). Mais, elle demande d'être effectuée à la main, en stockant les étapes intermédiaires, et d'effectuer la boucle de transduction jusqu'à ce que plus aucune nouvelle structure ne soit renvoyée. Nous pouvons imaginer un algorithme automatisant ces manipulations, mais ce n'est pas ce que nous recherchons ; cette alternative ne serait pas plus optimale. La transduction montre ici des limites que nous souhaitons dépasser dans nos recherches futures. Aussi, tous les types de paraphrases possibles (Iordanskaja *et al.*, 1991) n'ont pas été exploités et considérés ici : nous voulons continuer dans cette direction par la suite afin de couvrir cela. De plus, nous voulons réussir à ajouter les autres structures de la TST, au moins les structures communicatives des différents niveaux, afin d'avoir une implémentation encore plus conforme à la TST.

Remerciements

Je remercie Sylvain Pogodalla ainsi que les relecteurs pour leurs commentaires et remarques constructives, qui ont permis l'amélioration de cet article. Je tenais aussi à remercier mes deux directeurs de thèse, Philippe de Groote et Sylvain Pogodalla, pour leur encadrement et nos échanges qui m'ont permis d'écrire cet article.

Références

BANARESCU L., BONIAL C., CAI S., GEORGESCU M., GRIFFITT K., HERMJAKOB U., KNIGHT K., KOEHN P., PALMER M. & SCHNEIDER N. (2013). Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, p. 178–186, Sofia, Bulgaria : Association for Computational Linguistics.

BLOM C., DE GROOTE P., WINTER Y. & ZWARTS J. (2011). Implicit Arguments : Event Modification or Option Type Categories ? In M. ALONI, V. KIMMELMAN, F. ROELOFSEN, G. W. SASSOON, K. SCHULZ & M. WESTERA, Éds., *18th Amsterdam Colloquium on Logic, Language and Meaning*, volume 7218 de *Lecture Notes in Computer Science*, p. 240–250, Amsterdam, Netherlands : Springer. DOI : 10.1007/978-3-642-31482-7_25, HAL : hal-00763102.

CANDITO M.-H. & KAHANE S. (1998). Can the TAG derivation tree represent a semantic graph ? an answer in the light of meaning-text theory. In *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+4)*, p. 21–24, University of Pennsylvania : Institute for Research in Cognitive Science.

COUSIN M. (2022). Génération de texte avec les grammaires catégorielles abstraites et la théorie sens-texte. Mémoire de master, Grenoble INP Ensimag. HAL : hal-03942766.

DANLOS L., MASKHARASHVILI A. & POGODALLA S. (2014). Génération de textes : G-tag revisité avec les grammaires catégorielles abstraites. In B. BIGI, Éd., *Actes de TALN 2014 (Traitement automatique des langues naturelles)*, Marseille : ATALA LPL.

DE GROOTE P. (2001). Towards abstract categorial grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, p. 252–259, Toulouse, France : Association for Computational Linguistics. DOI : 10.3115/1073012.1073045.

IORDANSKAJA L., KITTREDGE R. & POLGUÈRE A. (1991). *Lexical Selection and Paraphrase in a Meaning-Text Generation Model*, In C. L. PARIS, W. R. SWARTOUT & W. C. MANN, Éds., *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, p. 293–312. Springer US : Boston, MA. DOI : 10.1007/978-1-4757-5945-7_11.

JARDINO M., Éd. (2005). *Actes de TALN 2005 (Traitement automatique des langues naturelles)*, Dourdan. ATALA, LIMSI.

KAHANE S. (2005). Structure des représentations logiques, polarisation et sous-spécification. In (Jardino, 2005).

KAHANE S. & LAREAU F. (2005). Grammaire d'unification sens-texte : modularité et polarisation. In (Jardino, 2005).

KANAZAWA M. (2007). Parsing and generation as datalog queries. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, p. 176–183, Prague, Czech Republic : Association for Computational Linguistics.

LAMBREY F. & LAREAU F. (2015). Le traitement des collocations en génération de texte multilingue. In *Actes de la 22e conférence sur le Traitement Automatique des Langues Naturelles. Articles courts*, p. 263–269, Caen, France : ATALA.

LAREAU F. (2007). Vers une formalisation des décompositions sémantiques dans la grammaire d'unification sens-texte. In F. BENAMARA, N. HATOUT, P. MULLER & S. OZDOWSKA, Éds., *Actes de TALN 2007 (Traitement automatique des langues naturelles)*, Toulouse : ATALA IRIT.

LAREAU F., LAMBREY F., DUBINSKAITE I., GALARRETA-PIQUETTE D. & NEJAT M. (2018). GenDR : A generic deep realizer with complex lexicalization. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan : European Language Resources Association (ELRA).

MEL'ČUK I. & POLGUÈRE A. (2021). Les fonctions lexicales dernier cri. In S. MARENGO, Éd., *La Théorie Sens-Texte. Concepts-clés et applications*, Dixit Grammatica, p. 75–155. L'Harmattan. HAL : hal-03311348.

MEL'ČUK I., MEL'ČUK I., BECK D. & POLGUÈRE A. (2012). *Semantics : From Meaning to Text*, volume 1 de *Semantics : From Meaning to Text*. John Benjamins Publishing Company.

MEL'ČUK I., MEL'ČUK I., BECK D. & POLGUÈRE A. (2013). *Semantics : From Meaning to Text*, volume 2 de *Semantics : From Meaning to Text*. John Benjamins Publishing Company.

MEL'ČUK I., MEL'ČUK I., BECK D. & POLGUÈRE A. (2015). *Semantics : From Meaning to Text*, volume 3 de *Semantics : From Meaning to Text*. John Benjamins Publishing Company.

MILIĆEVIĆ J. (2006). A short guide to the meaning-text linguistic theory. *Journal of Koralex*, **8**, 187–233.

POGODALLA S. (2016). ACGtk : un outil de développement et de test pour les grammaires catégorielles abstraites (ACG TK : a toolkit to develop and test abstract categorial grammars). In *Actes de la conférence conjointe JEP-TALN-RECITAL 2016. volume 5 : Démonstrations*, p. 1–2, Paris, France : AFCP - ATALA.

POGODALLA S. (2017a). A syntax-semantics interface for Tree-Adjoining Grammars through Abstract Categorial Grammars. *Journal of Language Modelling*, **5**(3), 527–605. DOI : 10.15398/jlm.v5i3.193, HAL : hal-01242154.

POGODALLA S. (2017b). Abstract Categorial Grammars as a Model of the Syntax-Semantics Interface for TAG. In *FSMNLP 2017 and TAG+13 conference*, Umeå, Sweden. HAL : hal-01583962.

RANTA A. (2004). Grammatical framework. *J. Funct. Program.*, **14**, 145–189. DOI : 10.1017/S0956796803004738.

WANNER L., BOHNET B., BOUAYAD-AGHA N., LAREAU F. & NICKLASS D. (2010). Marquis : Generation of user-tailored multilingual air quality bulletins. *Applied Artificial Intelligence*, **24**(10), 914–952. DOI : 10.1080/08839514.2010.529258.