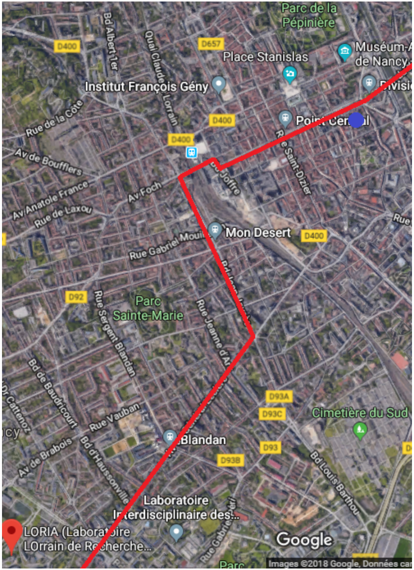
Traduction des grammaires catégorielles de Lambek dans les grammaires catégorielles abstraites

Valentin RICHARD
sous la direction de Philippe de Groote
Équipe SEMAGRAMME
Loria (Nancy)

du 11 juin au 27 juillet 2018

Nancy

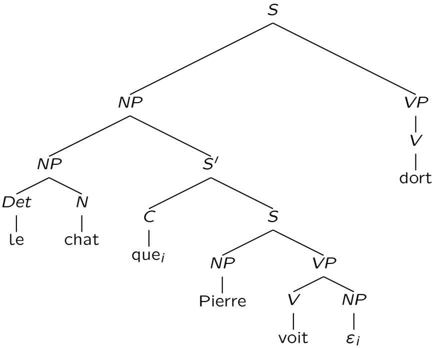


- Ligne 1 de tramway
- Mon lieu de résidence

Le Loria

- Unité Mixte de Recherche (UMR)
- 190 (enseignants-)chercheurs et 100 doctorants
- 27 équipes de recherche
- 3 grands bâtiments

La linguistique informatique et le traitement automatique des langues (TAL)



- Traduction automatique
- Recherche d'informations
- Résumé
- Modèles de la langue naturelle
- ...

- Outils de plus en plus demandés

L'équipe SEMMAGRAMME

- ACGtk : analyseur grammatical
- Outil d'aide au diagnostic de la schizophrénie
- Zombilingo : jeu pour l'annotation de ressources
- Sémantique à continuation et grammaires catégorielles

Mon maître de stage : Philippe de Groote

1 Environnement de travail

2 Sujet de recherche
- Système logique et grammaire de Lambek
- λ-calcul typé
- Grammaire catégorielle abstraite

3 Transformation des règles en coupure

Types orientés et grammaire de Lambek

Pour Pr un ensemble de **types atomiques**, **types orientés** Tp :

$$\mathrm{Tp} : \alpha, \beta, \gamma, \delta, \ldots ::= s, r, p, q, \ldots \in \mathrm{Pr} \mid \beta/\alpha \mid \alpha\backslash\beta$$

Grammaire de Lambek [Lam58] (LG) $\mathcal{G} = (\mathrm{Pr}, \mathcal{T}, \chi, s)$ où

- Pr types atomiques
- $\mathcal{T}$ symboles terminaux, appelés **symboles lexicaux** ou lexèmes
- $\chi : \mathcal{T} \to \mathcal{P}(\mathrm{Tp})$
- $s \in \mathrm{Tp}$ un type distingué (dans la suite $s \in \mathrm{Pr}$)

Système de dérivation et langage d'une LG

Système $\mathcal{S}_{IE}$ de dérivation formé sur les règles de séquents $\Gamma \vdash \beta$, avec Γ mot non vide sur $\mathrm{Tp} \cup \mathcal{T}$ et $\beta \in \mathrm{Tp}$:

$$\frac{}{\alpha \vdash \alpha}\ (\mathrm{Ax.}) \qquad \frac{\alpha \in \chi(t)}{t \vdash \alpha}\ (\mathrm{Lex.})$$

$$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \beta/\alpha}\ (/\mathrm{I}) \qquad \frac{\Gamma \vdash \beta/\alpha \qquad \Delta \vdash \alpha}{\Gamma, \Delta \vdash \beta}\ (/\mathrm{E})$$

$$\frac{\alpha, \Gamma \vdash \beta}{\Gamma \vdash \alpha\backslash\beta}\ (\backslash\mathrm{I}) \qquad \frac{\Gamma \vdash \alpha\backslash\beta \qquad \Delta \vdash \alpha}{\Delta, \Gamma \vdash \beta}\ (\backslash\mathrm{E})$$

$$\frac{\Phi, \gamma, \Delta \vdash \beta \qquad \Gamma \vdash \gamma}{\Phi, \Gamma, \Delta \vdash \beta}\ (\mathrm{CUT})$$

Figure – Règles du calcul de Lambek (déduction naturelle) sans produit lexicalisé $\mathcal{S}_{IE}$

- **Dérivation** (ou preuve) : arbre dont les nœuds sont des instances des règles
- **Langage** de la grammaire de Lambek : $\mathcal{L}(\mathcal{G}) \triangleq \{\Gamma \in \mathcal{T}^+ \mid \Gamma \vdash_{IE} s\}$
- Analyse grammaticale (parsing) = recherche de preuve

Exemple

Grammaire $\mathcal{G}_0$ définie sur Pr $\triangleq \{s, n, np\}$

LE :	np/n	CHAT :	n	DORT :	$np\backslash s$
QUE :	$(n\backslash n)/(s/np)$	PIERRE :	np	VOIT :	$(np\backslash s)/np$

Table – Exemple 0 : Grammaire de Lambek $\mathcal{G}_0$

Exemple

Grammaire $\mathcal{G}_0$ définie sur Pr $\triangleq \{s, n, np\}$

LE :	np/n	CHAT :	n	DORT :	$np\backslash s$
QUE :	$(n\backslash n)/(s/np)$	PIERRE :	np	VOIT :	$(np\backslash s)/np$

Table – Exemple 0 : Grammaire de Lambek $\mathcal{G}_0$

$$\dfrac{\dfrac{Q \vdash (n\backslash n)/(s/np) \qquad \dfrac{\dfrac{\dfrac{V \vdash (np\backslash s)/np \qquad np \vdash np}{V, np \vdash np\backslash s}(/E) \qquad P \vdash np}{P, V, np \vdash s}(\backslash E)}{P, V \vdash s/np}(/I)}{Q, P, V \vdash n\backslash n}(/E) \qquad C \vdash n}{C, Q, P, V \vdash n}(\backslash E)$$

Figure – Exemple 0 : Haut de la dérivation de LE CHAT QUE PIERRE VOIT DORT de $\mathcal{G}_0$ dans $\mathcal{S}_{IE}$

λ-calcul linéaire avec constantes

Pour $\mathcal{T}$ considéré comme un ensemble de constantes et $\mathcal{X}$ un ensemble infini dénombrable de variables, **λ-termes** linéaire $\Lambda(\mathcal{T})$:

$$u, v, w \ldots ::= x, y, z \ldots \in \mathcal{X} \mid t, \ldots \in \mathcal{T} \mid \lambda x.\, u \; (x \textit{ au plus une fois libre dans } u) \mid u\, v$$

- Convention de parenthésage $u\, v\, w \triangleq (u\, v)\, w$
- Typage simple (non orienté) sur un ensemble de types de base $\mathcal{B}$:

$$\mathrm{Tp}_{no}(\mathcal{B}) : \alpha, \beta, \gamma, \delta, \ldots ::= s, r, p, q, \ldots \in \mathcal{B} \mid \alpha \to \beta$$

$$\frac{}{x : \alpha \vdash x : \alpha}\,(\mathrm{Ax.}_{no}) \qquad \frac{\alpha \in \chi_{no}(t)}{\vdash t : \alpha}\,(\mathrm{Lex.}_{no})$$

$$\frac{\Gamma, x : \alpha \vdash u : \beta}{\Gamma \vdash \lambda x.\, u : \alpha \to \beta}\,(\to\mathrm{I}_{no}) \qquad \frac{\Gamma \vdash u : \alpha \to \beta \qquad \Delta \vdash v : \alpha}{\Gamma, \Delta \vdash u\, v : \beta}\,(\to\mathrm{E}_{no})$$

Figure – Règles de typage simple (non orienté) $\mathcal{S}_{no}$ du λ-calcul linéaire avec constantes

Exemple : $D\,(L\,(Q\,(\lambda x.\, V\, x\, P)\; C)) \vdash s$

Signature d'ordre supérieure et lexique

Signature d'ordre supérieur $\Sigma = (\mathcal{B}, C, \tau)$ où

- $\mathcal{B}$ ensemble fini de types de base
- C ensemble fini de constantes
- $\tau : C \to \mathrm{Tp}_{no}(\mathcal{B})$

Signature d'ordre supérieure et lexique

Signature d'ordre supérieur $\Sigma = (\mathcal{B}, C, \tau)$ où

- $\mathcal{B}$ ensemble fini de types de base
- C ensemble fini de constantes
- $\tau : C \to \mathrm{Tp}_{no}(\mathcal{B})$

Lexique de Σ_1 vers Σ_2 : un morphisme de signature d'ordre supérieur $\mathfrak{L} = (F, G)$ où

- $F : \mathcal{B}_1 \to \mathrm{Tp}_{no}(\mathcal{B}_2)$
- $G : C_1 \to \Lambda(C_2)$
- pour tout $c \in C_1$, on a $\vdash_{no} G(c) : F(\tau_1(c))$

Grammaire catégorielle abstraite

Grammaire catégorielle abstraite [dG01] $\mathfrak{G} = (\Sigma_1, \Sigma_2, \mathfrak{L}, s)$ où

- Σ_1 (resp. Σ_2) une signature d'ordre supérieur appelé le **vocabulaire abstrait** (resp. **objet**)
- $\mathfrak{L} : \Sigma_1 \to \Sigma_2$ un lexique de Σ_1 vers Σ_2.
- $s \in \mathrm{Tp}_{no}(\mathcal{B}_1)$ un type distingué

Langage objet : $\mathfrak{O}(\mathfrak{G}) \triangleq \{v \in \Lambda(C_2) \mid \exists u \in \Lambda(C_1), \vdash_{no} u : s \wedge v = \hat{G}(u)\}$

$L : [n] \to [np]$	$\mapsto \lambda x.\, LE + x$	$C : [n]$	$\mapsto CHAT$
$D : [np] \to [s]$	$\mapsto \lambda x.\, x + DORT$	$P : [np]$	$\mapsto PIERRE$

Table – Exemple d'ACG sous la forme $c_1 : \tau(c_1) \mapsto \mathfrak{L}(c_1)$

Grammaire algébrique

Grammaire algébrique (CFG) $\mathcal{G}_A = (N, A, P, s)$ où

- N alphabet fini de symboles non terminaux $(K, L, \ldots)$
- A alphabet fini de symboles terminaux $(a, b, \ldots)$, $(N \cap A = \emptyset)$
- $P \in \mathcal{P}_f(N \times (N \cup A)^*)$ ensemble fini de productions notées $K \longrightarrow w$
- $s \in N$ un symbole distingué

Grammaire algébrique

Grammaire algébrique (CFG) $\mathcal{G}_A = (N, A, P, s)$ où

- N alphabet fini de symboles non terminaux $(K, L, \ldots)$
- A alphabet fini de symboles terminaux $(a, b, \ldots)$, $(N \cap A = \emptyset)$
- $P \in \mathcal{P}_f(N \times (N \cup A)^*)$ ensemble fini de productions notées $K \to w$
- $s \in N$ un symbole distingué

munie d'un système de réécriture de règle le passage au contexte de P :

$$\frac{K \to w \quad \in P}{w_1 K w_2 \Rightarrow w_1 w w_2}.$$

- $\Rightarrow^*$: la clôture réflexive transitive de $\Rightarrow$
- **langage** de la grammaire algébrique : $\mathcal{L}(\mathcal{G}_A) \triangleq \{w \in A^* \mid s \Rightarrow^* w\}$
- un symbole non terminal K est **accessible** s'il existe une dérivation $s \Rightarrow^* w_1 K w_2$

Exemple pour P : $s \to np, DORT$; $np \to LE, n$ et $n \to CHAT$:

$$s \Rightarrow np, DORT \Rightarrow LE, n, DORT \Rightarrow LE, CHAT, DORT$$

Une première traduction insatisfaisante

- Traduction naïve des types orientés en types non orientés :

$$\begin{aligned} \tau_0(r) &= r && \text{si } r \in \mathrm{Pr} \\ \tau_0(\beta/\alpha) &= \tau_0(\alpha) \to \tau_0(\beta) \\ \tau_0(\alpha\backslash\beta) &= \tau_0(\alpha) \to \tau_0(\beta) \end{aligned}$$

et λ-termes associés à la preuve. Mais **surgénère** ! ex :
QUE: $(np \to s) \to n \to n$ indifférentiable de QUI: $(np \to s) \to n \to n$

Une première traduction insatisfaisante

- Traduction naïve des types orientés en types non orientés :

$$\begin{aligned} \tau_0(r) &= r && \text{si } r \in \mathrm{Pr} \\ \tau_0(\beta/\alpha) &= \tau_0(\alpha) \to \tau_0(\beta) \\ \tau_0(\alpha\backslash\beta) &= \tau_0(\alpha) \to \tau_0(\beta) \end{aligned}$$

  et λ-termes associés à la preuve. Mais **surgénère** ! ex :
  QUE: $(np \to s) \to n \to n$ indifférentiable de QUI: $(np \to s) \to n \to n$

- En utilisant le résultat de Pentus [Pen97] : les langages des grammaires de Lambek sont exactement les **langages algébriques**

Problème : Beaucoup de séquents inutiles ! exemple :
VOIT: $(n \to np) \to np \to n \to s$ dans (PIERRE VOIT LE) CHAT

Une première traduction insatisfaisante

- Traduction naïve des types orientés en types non orientés :

$$\begin{aligned} \tau_0(r) &= r & \text{si } r \in \mathrm{Pr} \\ \tau_0(\beta/\alpha) &= \tau_0(\alpha) \to \tau_0(\beta) \\ \tau_0(\alpha\backslash\beta) &= \tau_0(\alpha) \to \tau_0(\beta) \end{aligned}$$

  et λ-termes associés à la preuve. Mais **surgénère** ! ex :
  QUE: $(np \to s) \to n \to n$ indifférentiable de QUI: $(np \to s) \to n \to n$

- En utilisant le résultat de Pentus [Pen97] : les langages des grammaires de Lambek sont exactement les **langages algébriques**

Problème : Beaucoup de séquents inutiles ! exemple :
VOIT: $(n \to np) \to np \to n \to s$ dans (PIERRE VOIT LE) CHAT

Idée :

- Garder le principe des grammaires algébriques : par les **coupures**
- Transformation par réécriture de preuve

Suppression des règles d'élimination

- **Aplatissement** du type orienté comme mot autour du symbole lexical : variables pour coupures. ex : $n, Q, s/np \vdash n$
- de même pour certaines variables

$$
\dfrac{
  \dfrac{
    n, Q, s/np \vdash n
    \qquad
    \dfrac{
      \dfrac{
        \dfrac{
          np, V, np \vdash s \qquad np \vdash np
        }{
          np, V, np \vdash s
        }\,(\mathrm{CUT})
        \qquad
        P \vdash np
      }{
        P, V, np \vdash s
      }\,(\mathrm{CUT})
    }{
      P, V \vdash s/np
    }\,(/\mathrm{I})
  }{
    n, Q, P, V \vdash n
  }\,(\mathrm{CUT})
  \qquad
  C \vdash n
}{
  C, Q, P, V \vdash n
}\,(\mathrm{CUT})
$$

- Ensemble des axiomes créés : $\mathcal{A}_0$

Suppression des règles d'introduction

- Réécriture $\rightsquigarrow$ en intégrant les introductions (abstraction) directement dans les axiomes :

$$\dfrac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \beta/\alpha}\ (/\mathrm{I}) \quad \rightsquigarrow \quad \Gamma \vdash \beta/\alpha$$

(idem pour \)

- Axiomes construits par induction $\mathcal{A}_2 ::= \mathcal{A}_0 \mid \mathcal{Q}_1(\mathcal{A}_2) \mid \mathcal{Q}_2(\mathcal{A}_2)$, avec les **opérateurs** $\mathcal{Q}_1$ et $\mathcal{Q}_2$

$\mathcal{Q}_1(\mathcal{A})$: si $\Gamma, \gamma \vdash \beta \in \mathcal{A}$ et $\alpha \vdash_{IE} \gamma$ alors ajouter $\Gamma \vdash \beta/\alpha$

$\mathcal{Q}_2(\mathcal{A})$: si $\Gamma, \alpha \vdash \beta \in \mathcal{A}$ alors ajouter $\Gamma, \alpha/\delta \vdash \beta/\delta$
(idem pour \)

Suppression des règles d'introduction

- Réécriture $\rightsquigarrow$ en intégrant les introductions (abstraction) directement dans les axiomes :

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma, \alpha \vdash \beta\end{array}}{\Gamma \vdash \beta/\alpha}\,(/\mathrm{I}) \quad \rightsquigarrow \quad \begin{array}{c}\vdots\\ \Gamma \vdash \beta/\alpha\end{array} \qquad \text{(idem pour } \backslash\text{)}$$

- Axiomes construits par induction $\mathcal{A}_2 ::= \mathcal{A}_0 \mid \mathcal{Q}_1(\mathcal{A}_2) \mid \mathcal{Q}_2(\mathcal{A}_2)$, avec les **opérateurs** $\mathcal{Q}_1$ et $\mathcal{Q}_2$

$\mathcal{Q}_1(\mathcal{A})$: si $\Gamma, \gamma \vdash \beta \in \mathcal{A}$ et $\alpha \vdash_{IE} \gamma$ alors ajouter $\Gamma \vdash \beta/\alpha$

$\mathcal{Q}_2(\mathcal{A})$: si $\Gamma, \alpha \vdash \beta \in \mathcal{A}$ alors ajouter $\Gamma, \alpha/\delta \vdash \beta/\delta$ (idem pour \)

$$\dfrac{\dfrac{n, Q, s/np \vdash n \qquad \dfrac{np, V \vdash s/np \qquad P \vdash np}{P, V \vdash s/np}\,(\mathrm{CUT})}{n, Q, V, P \vdash n}\,(\mathrm{CUT}) \qquad C \vdash n}{C, Q, P, V \vdash n}\,(\mathrm{CUT})$$

Des coupures aux ACG

- Grammaire algébrique $\mathcal{G}_A \triangleq (N, \mathcal{T}, P, s)$ où N est l'ensemble des symboles lexicaux et types de $\mathcal{A}_2$, et

$$P \triangleq \{\beta \to \Gamma \mid \Gamma \vdash \beta \in \mathcal{A}_2\}$$

- Traduction d'une grammaire algébrique en ACG par [dGPog04] avec vocabulaire objet des chaînes de caractères
- Exemple :

$L : [n] \to [np]$	$\mapsto \lambda x.\, LE + x$	$C : [n]$	$\mapsto CHAT$
$D : [np] \to [s]$	$\mapsto \lambda x.\, x + DORT$	$Q : [s/np] \to [n] \to [n]$	$\mapsto \lambda x.\, \lambda y.\, y + QUE + x$
$P : [np]$	$\mapsto PIERRE$	$V : [np] \to [s/np]$	$\mapsto \lambda y.\, y + VOIT$

Table – Exemple 0 : Version finale de $\mathcal{G}_0$ en ACG

Algorithme de mise à niveau

- **Problème :** $\mathcal{A}_2$ potentiellement infini

→ **Algorithme** : création d'axiomes arrêtée à un certain niveau
  - polynomial en la grammaire initial
- En pratique, quelques itérations des opérateurs suffisent
  - introduction = extraction linguistique : PIERRE QUI MANGE LA POMME
- Amélioration possible : travailler en accessibilité (au type s)

Conclusion

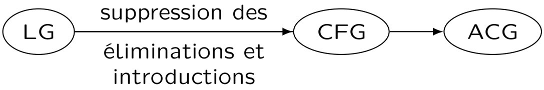


Figure – Schéma de la traduction d'une LG en ACG

Mon apport :

- Écriture et **démonstration** du procédé
- Ébauche de l'algorithme

Conclusion

Merci pour votre attention

Annexes

Types de coupures :

$$\frac{\begin{array}{c} | \\ \Phi, \gamma, \Delta, \alpha \vdash \beta \end{array} \qquad \begin{array}{c} | \\ \Gamma \vdash \gamma \end{array}}{\Phi, \Gamma, \Delta, \alpha \vdash \beta} \text{(CUT)}$$

Cas 1

$$\frac{\begin{array}{c} | \\ \Phi, \gamma \vdash \beta \end{array} \qquad \begin{array}{c} | \\ \Gamma, \alpha \vdash \gamma \end{array}}{\Phi, \Gamma, \alpha \vdash \beta} \text{(CUT)}$$

Cas 2

(idem pour α à gauche)

Figure – Cas de la forme d'une coupure pour la suppression d'une introduction