Méta-linguistique dans l'assistant de preuves Coq

Houda Anoun

Méta-linguistique dans l'assistant de preuves Coq

Houda Anoun
LaBRI - Bordeaux 1,
351, Rue de la libération,
33405, Talence, France
anoun@labri.fr

Résumé : Les grammaires catégorielles sont des formalismes très généraux basés sur des systèmes d'inférence logiques et dédiés à l'analyse des langues naturelles. Elles ont l'avantage de garantir une interface syntaxe/sémantique aisée. Plusieurs modèles logiques enrichis ont été récemment définis pour rendre compte de différents phénomènes linguistiques assez complexes. Toutefois, l'adéquation de ces formalismes vient au détriment de leur simplicité et leur complexité d'analyse. Nous réalisons un outil destiné à la recherche et l'enseignement de la linguistique computationnelle, permettant d'appréhender ces modèles, en construisant des analyses syntaxiques de phrases et leur interprétation sémantique, mais aussi en établissant des propriétés de classes entières de langages et en étudiant les relations entre divers formalismes. Cet outil : l'atelier $\mathcal{ICHARATE}$, est composé de bibliothèques pour l'assistant à la démonstration Coq. Nous décrivons dans cet article les principales fonctionnalités de cet atelier, en montrant comment il répond aux besoins exprimés ci-dessus.

Mots-clés : Traitement des langues naturelles, grammaires catégorielles, interface syntaxe/sémantique, logique d'ordre supérieur, assistants de preuves.

1 SYSTÈMES DÉDUCTIFS ET LINGUISTIQUE COMPUTATIONNELLE

1.1 Pourquoi la logique ?

Les systèmes déductifs ont été introduits au début du vingtième siècle pour formaliser entièrement différents domaines et éviter les paradoxes émanant des définitions intuitives et informelles (e.g. ensemble des ensembles et paradoxe de Russel).

Depuis les années soixantes, un effort continu est mené pour automatiser le plus possible la démonstration de théorèmes sur ordinateur. Cette activité permet de vérifier les propriétés de modèles complexes, notamment dans le domaine de transport, de la cryptologie, des télécommunications etc.

La modélisation de la syntaxe et de la sémantique des langues naturelles fait appel à des formalismes caractérisés par un grand degré de paramétrisation. L'étude des propriétés de ces modèles peut donc bénéficier de la mécanisation du raisonnement. Parmi les formalismes utilisés dans l'étude des langues naturelles nous nous intéressons aux grammaires catégorielles qui sont basées sur des systèmes déductifs logiques.

Dans les langues naturelles, l'ordre et le nombre des mots d'une phrase sont d'une grande importance, nous avons donc besoin d'une logique sensible aux ressources qui prend en compte aussi bien le nombre que l'ordre des hypothèses dans le contexte, à la différence des logiques classiques et intuitionnistes.

Joachim Lambek fut le premier à définir un système déductif complet [Lambek, 1961] qui vérifie ces contraintes. Son système, nommé **L**, est une variante non commutative de la logique linéaire [Girard, 1987].

Pour prendre en charge l'ordre des mots, la flèche fonctionnelle de la logique intuitionniste est scindée en deux connecteurs $/$ et $\backslash$. Le type A$/$B (resp. B$\backslash$A) est ainsi associé à toute entité qui attend une expression de type B à sa droite (resp. gauche) pour former une expression de type A. Le connecteur $\bullet$, quant à lui, peut être vu comme le dual de la famille ($/$, $\backslash$) : une expression est de type A$\bullet$B si elle résulte de la concaténation de deux entités de types respectifs A et B. L'ensemble des types manipulés par notre logique est le plus petit ensemble qui comprend les types atomiques représentant les expressions complètes (e.g. **n** : nom commun, **s** : phrase correcte, **np** : syntagme nominal) et qui est clos par rapport aux trois connecteurs $/$, $\backslash$ et $\bullet$. Ces types permettent de décrire les différents comportements syntaxiques des mots. A titre d'exemple, on associe au verbe *mange* les deux types **np**$\backslash$**s** et **(np**$\backslash$**s)**$/$**np** étant donné que ce dernier peut être, selon les cas, soit transitif (e.g. *Houda mange une pomme*) soit intransitif (e.g. *Houda mange*). En revanche, la sémantique des mots est représentée par des λ-termes simplement typés où l'on se restreint aux deux types de base *e* (individus) et *t* (valeurs de vérité) [Montague, 1974, GAMUT, 1991]. Un morphisme de types noté $()^*$ permet de transformer les types syntaxiques en des types sémantiques et ce de manière compositionnelle (i.e. $(A/B)^*$=$(B\backslash A)^*$=B$^*\rightarrow A^*$, $(A \bullet B)^*$=A$^* \times B^*$). La fonction d'assignation qui attribue à chaque mot un ensemble fini de types syntaxiques et de représentations sémantiques est nommée le *lexique*.

A la différence des grammaires hors-contexte (CFG), les grammaires catégorielles ont un ensemble de règles de production indépendant des terminaux (i.e. mots de la langue choisie dans notre cas). En outre, elles garantissent une interface syntaxe/sémantique aisée grâce à la correspondance de Curry-Howard (correspondance entre

les preuves constructives et les λ-termes). La figure 1 montre deux exemples de règles d'inférence du système **L** ainsi que l'interface syntaxe/sémantique associée.

Syntaxe	Sémantique
$\dfrac{\Gamma \vdash A/B \quad \Delta \vdash B}{\Gamma, \Delta \vdash A}\ /E$	$\dfrac{\Gamma^* \vdash f : B^* \to A^* \quad \Delta^* \vdash b : B^*}{\Gamma^*, \Delta^* \vdash (f\ b) : A^*} \to E$
$\dfrac{\Gamma, x : B \vdash A}{\Gamma \vdash A/B}\ /I$	$\dfrac{\Gamma^*, x^* : B^* \vdash f : A^*}{\Gamma^* \vdash \lambda x^*.f : B^* \to A^*} \to I$

FIG. 1 – Interface Syntaxe/Sémantique dans L

Dans les règles de la figure 1, les lettres majuscules latines (e.g. A, B) représentent des types syntaxiques, les lettres grecques (e.g. Γ, Δ), quant à elles, symbolisent des contextes ordonnés de types syntaxiques préfixés par des variables formelles et $\vdash$ est l'opérateur de dérivation. Remarquons qu'à chaque étape de déduction correspond une étape de calcul de la sémantique dans le λ-calcul simplement typé (e.g. l'application pour la règle /E et l'abstraction pour /I).
Les dérivations syntaxiques de phrases peuvent être vues comme des preuves de théorèmes dans les systèmes logiques considérés. En effet, une phrase $m_1 m_2 ... m_n$ appartient au langage engendré par notre grammaire ssi pour chaque mot m_i il existe un type t_i qui lui est associé par le lexique tel que ($t_1 t_2 ... t_n \vdash$ s) soit démontrable dans le système déductif sous-jacent.

1.2 Exemple

Illustrons le fonctionnement des grammaires de Lambek sur un exemple simple.
On considère la grammaire rigide (i.e. dont le lexique attribue à chaque mot un unique type syntaxique) définie par le lexique de la figure 2 :

Houda	*lit*	*roman*	*Amok*	*dort*
np	(np\s)/np	n	np	np\s
e	e→e→t	e→ t	e→ t	e→ t
Houda	**Lire**	**Roman**	**Amok**	**Dormir**

le	*que*
(s/(np\s))/n	(n\n)/(s/np)
(e→ t)→ (e→ t)→ t	(e→ t)→ (e→ t)→ e→ t
λP λQ ∃ !x. P(x) ∧ Q(x)	λP λQ λx. P(x)∧ Q(x)

qui	*s'intitule*	*et*
(n\n)/(np \ s)	(np\s)/np	(s\s)/s
(e→ t)→ (e→ t)→ e→ t	e→e→ t	t→ t→ t
λP λQ λx. P(x)∧ Q(x)	**Intituler**	λx λy. x∧ y

FIG. 2 – Lexique 1

Le lexique ci dessus associe à chaque mot, un type syntaxique, un type sémantique et un λ-terme simplement typé qui représente sa sémantique. Les constantes sémantiques (individus et prédicats) sont imprimées en gras. Par exemple, la constante **Roman** est un prédicat à un seul argument représentant l'ensemble des propriétés d'un roman. Certains mots, dont les quantificateurs (e.g. *le, un, tout*) et les pronoms relatifs (e.g. *que, qui*) ont une sémantique logique qui ne contient aucune constante sémantique et dont l'interprétation reste invariable d'un modèle à un autre. Notons au passage l'utilisation de la catégorie d'ordre supérieur **(s/(np\s))/n** pour typer l'article défini *'le'*. A la différence du type simple **np**/ **n**, notre catégorie nous permet d'attribuer une sémantique logique à cet article.
A partir du lexique de la figure 2, on peut montrer par exemple que la phrase *'Le roman que Houda lit s'intitule Amok'* est syntaxiquement correcte (i.e. reconnaissable par la grammaire). Les deux étapes principales de la dérivation sont représentées dans la figure 3. Pour plus de lisibilité, on manipule des contextes composés de ressources linguistiques dont les types sont implicites, ceci ne pose aucune ambiguïté car notre grammaire est rigide.

$$\dfrac{\dfrac{}{le, ..., lit \vdash s/(np\backslash s)} \quad \dfrac{\dfrac{}{s'intitule \vdash (np\backslash s)/np}\ Ax \quad \dfrac{}{Amok \vdash np}\ Ax}{s'intitule,\ Amok \vdash np\backslash s}\ /E}{le,\ roman,\ que,\ Houda,\ lit,\ s'intitule,\ Amok \vdash s}\ /E$$

$$\dfrac{\dfrac{}{Houda \vdash np}\ Ax \quad \dfrac{\dfrac{}{lit \vdash (np\backslash s)/np}\ Ax \quad \dfrac{}{x : np \vdash np}\ Ax}{lit,\ (x : np) \vdash np\backslash s}\ /E}{\dfrac{\dfrac{\dfrac{Houda,\ lit,\ (x : np) \vdash s}{Houda,\ lit \vdash s/np}\ /I}{que,\ Houda,\ lit \vdash n\backslash n}\ /E}{\vdots}}{le,\ roman,\ que,\ Houda,\ lit \vdash s/(np\backslash s)}\ /E$$

FIG. 3 – Exemple de dérivation en L

La dérivation syntaxique utilise les règles d'introduction et d'élimination des deux connecteurs / et \ (cf. figure 1) ainsi que la règle d'axiome (x :A ⊢ A). Si on visualise la dérivation comme un arbre, alors la racine est le séquent qu'on souhaite analyser et les feuilles sont les axiomes logiques. La partie délicate de cette preuve est celle qui permet d'assigner à la subordonnée relative *'que Houda lit'* le type **n\n**. Pour atteindre cet objectif, il suffit de prouver que *'Houda lit'* porte le type **s/np**. Ceci se fait en appliquant la règle d'introduction de / qui fait recours au raisonnement hypothétique. Une hypothèse 'x', de type **np**, est donc introduite dans le contexte et placée à l'extrême droite de ce dernier. Il suffit ainsi de prouver que *'Houda lit x'* est une phrase correcte, chose qui se fait assez aisément.
Par la correspondance de Curry-Howard, on construit facilement un λ-terme représentant la sémantique dérivationnelle de notre phrase et ce à partir de son arbre de dérivation syntaxique. La figure 4 montre la dérivation du jugement de typage permettant de construire la sémantique associée à notre phrase relative[1].
La sémantique finale de notre phrase est obtenue en substituant chaque variable formelle 'w^*' par la

[1]Dans cette d´erivation, chaque variable formelle de la forme 'w^*' repr´esente la s´emantique du mot 'w'

$$\dfrac{\dfrac{\dfrac{\dfrac{}{Houda^* \vdash Houda^* : e}\,Ax \quad \dfrac{\dfrac{}{lit^* \vdash lit^* : e \to e \to t}\,Ax \quad \dfrac{}{x : e \vdash x : e}\,Ax}{lit^*,\ (x : e) \vdash lit^*(x) : e \to t} \to E}{Houda^*,\ lit^*,\ (x : e) \vdash lit^*(x, Houda^*) : t} \to E}{Houda^*,\ lit^* \vdash \lambda x.lit^*(x, Houda^*) : e \to t} \to I}{que^*,\ Houda^*,\ lit^* \vdash que^*(\lambda x.lit^*(x, Houda^*)) : (e \to t) \to e \to t} \to E$$

FIG. 4 – Calcul de la sémantique

sémantique lexicale du mot qu'elle représente, et ce dans la sémantique dérivationnelle de cette phrase. Cette sémantique est représentée dans la logique des prédicats de premier ordre comme ceci :
$\exists$!x. **Lire**(x, **Houda**)$\wedge$**Roman**(x)$\wedge$**Intituler**(**Amok**, x)

1.3 Vers une logique multimodale enrichie

Le système *L*, malgré son élégance et sa simplicité, ne s'avère pas très adapté à l'analyse des langues naturelles pour diverses raisons. Dans un premier temps, ce système a une capacité générative très limitée : en effet, il a été prouvé que les langages engendrés par ces grammaires sont hors-contextes ; en revanche, Noam Chomsky a conjecturé que les langues naturelles font partie des langages faiblement contextuels [Chomsky, 1957]. Plusieurs phénomènes linguistiques ne peuvent être traités dans ce système tels l'ellipse (omission d'une partie de la phrase pour alléger la formulation, e.g. *'Houda aime les mathématiques et Soumaya _ la médecine'*) et l'extraction médiane (extraction d'un élément dont la position n'est pas périphérique dans un constituant linguistique e.g. *'Le roman que Houda lit _ souvent s'intitule Amok'*).
Le deuxième problème dont souffrent les grammaires de **Lambek** est le problème de sur-génération résultant de l'associativité non contrôlée de l'opérateur ',' qui regroupe les ressources dans le contexte. En effet, on peut facilement vérifier que certaines phrases qui violent la contrainte de la structure de coordination [Ross, 1967] (i.e. contrainte selon laquelle les deux membres d'une coordination doivent avoir le même type) sont dérivables dans **L**. Ceci est le cas de la phrase agrammaticale *(*Le roman que Houda dort et lit s'intitule Amok*). Afin de pallier ces faiblesses, Michael Moortgat [Moortgat, 1997] a défini un système logique enrichi par l'introduction de deux nouveaux connecteurs logiques unaires ($\Box$, $\Diamond$) vérifiant la règle ($\Diamond\Box$ A$\vdash$ A). En outre, ce système manipule une famille de connecteurs logiques indexés par des modes de composition ($/_i$, $\backslash_i$,$\bullet_i$, $\Box_j$,$\Diamond_j$), c'est pour cette raison que la logique sous-jacente est dite *multimodale*.
Les contextes, dans ce système multimodal, ne sont plus de simples listes plates mais plutôt des arbres binaires structurés où chaque noeud interne coïncide soit avec l'opérateur structurel binaire $(,)^i$, soit avec l'opérateur structurel unaire $<>_j$. Les contextes arborescents sont plus informatifs que les simples listes de ressources linguistiques car ils permettent d'encapsuler la structure de la phrase. La figure 5 représente un exemple de contexte dans la logique multimodale.
Une suite de mots *l* (l=$m_1 ... m_n$) porte le type **ty** dans une grammaire multimodale (on note ceci par l : **ty**) ssi pour

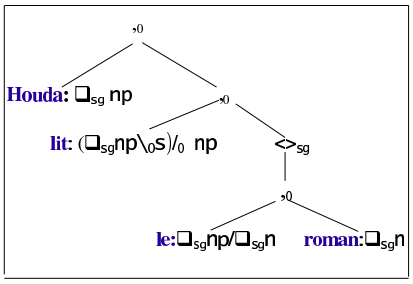

FIG. 5 – Exemple de contextes structurés

chaque mot m_i de l'expression *l* il existe un type t_i qui lui est assigné par le lexique et il existe un contexte structuré Γ dont les feuilles sont respectivement t_1, ...,t_n tels que le séquent ($\Gamma \vdash$ ty) soit dérivable dans la logique considérée.

Les connecteurs unaires servent, entre autre, à coder les traits morphosyntaxiques. On peut ainsi attribuer au nom commun *roman* le type enrichi $\Box_{sg}\ \Box_{ma}$ **n** pour préciser qu'il s'agit d'un nom commun masculin et singulier. De la même façon, on associe à l'article défini *'le'* la catégorie **np**$/\ \Box_{sg}\ \Box_{ma}$ **n** pour le forcer à se combiner avec des noms communs masculins et singuliers. On peut ainsi prouver dans cette logique que *'le roman'* est bien un groupe nominal alors que *'le romans'* ne l'est pas.
Les règles logiques ne suffisent pas pour rendre compte des différents phénomènes de la langue naturelle. Afin d'augmenter le pouvoir expressif d'une grammaire multimodale, on rajoute un ensemble de règles structurelles (i.e. règles de réécriture) permettant la gestion des ressources dans le contexte. Ces règles structurelles sont appliquées de manière contrôlée en présence de modes spécifiques pour éviter le problème de sur-génération. A titre d'exemple, on peut appliquer la commutativité uniquement lorsque l'opérateur ',' est décoré par le mode de composition **c** (cf. Fig. 6).

$$\dfrac{\Gamma[(\Delta_2\ ,\ \Delta_1)^c] \vdash C}{\Gamma[(\Delta_1\ ,\ \Delta_2)^c] \vdash C}\ P(c)$$

FIG. 6 – Règle structurelle de commutativité locale

Remarquons que cette règle fait intervenir une construction de la forme $\Gamma[\Delta]$. Dans ce cas, 'Γ[ ]' représente un contexte linéaire [Huet, 1997] qui définit l'emplacement où sera appliquée la règle de réécriture.
On peut appliquer cette règle pour rendre compte de la flexibilité des adjectifs dans un certain nombre de langues telles le français ou l'italien. Dans ces dernières langues, un grand nombre d'adjectifs peuvent se placer indifféremment à gauche ou à droite du nom commun qu'ils caractérisent. A titre d'exemple, en assignant à l'adjectif *'long'* le type **n**$/_c$**n**, on est capable de dériver les deux groupes nominaux *'un long roman'* et *'un roman long'* dans toute grammaire multimodale supportant la règle de la figure 6.
Il est possible d'augmenter la capacité générative des

grammaires multimodales en définissant des règles structurelles enrichies dites *principes d'interaction* et qui permettent la communication entre différents modes de composition. Diverses règles ont été définies de manière contrôlée pour permettre l'analyse de phénomènes complexes de la langue naturelle [Moortgat, 1997] ; citons à titre d'exemple le phénomène de dépendances parasites (i.e. dépendances sémantiques entre un nombre quelconque d'éléments d'une phrase) [Steedman, 1996] et qui est présent dans la phrase *'The man whom I will persuade the friends of _ to vote for _ is nice'*. En effet, dans cette phrase l'objet de l'infinitif *'to vote for'* et le complément de *'the friends of'* correspondent au même individu qui coincide avec l'élément extrait par le pronom relatif *'whom'*. Un tel phénomène peut être analysé dans le cadre des grammaires multimodales en ayant recours à une forme contrôlée de contraction illustrée dans la figure 7.

$$\frac{\Gamma[((\Delta_1, \Delta_3)^j, (\Delta_2, \Delta_3)^j)^i] \vdash C}{\Gamma[((\Delta_1, \Delta_2)^i, \Delta_3)^j] \vdash C}\ MC(i,j)$$

FIG. 7 – Règle de contraction contrôlée

1.4 Rôle des assistants de preuves

Les grammaires multimodales sont caractérisées par leur grand degré de paramétrisation (i.e. toute grammaire est paramétrée par l'ensemble des atomes, les modes, les mots, le lexique, les règles structurelles ...). Ceci rend leur compréhension et utilisation relativement ardues pour les novices. Il serait donc intéressant d'avoir des outils qui facilitent cette tâche aux utilisateurs. Le problème de recherche de démonstration dans la logique multimodale de Moortgat étant indécidable, on ne peut envisager de réaliser des analyseurs totalement automatiques qui prennent en charge toutes les classes de grammaires. Toutefois, il existe un nombre de prouveurs automatiques basés sur des classes particulières de logique multimodale qui imposent certaines contraintes sur les règles structurelles pour se garantir la décidabilité de l'analyse. Parmi ces outils citons l'analyseur syntaxique **Grail** réalisé par Richard Moot [Moot, 2002]. Cet analyseur est certes automatique, mais ne permet guère de raisonner sur le modèle sous-jacent en prouvant des propriétés génériques vérifiées par des classes entières de grammaires. En effet, on ne peut effectuer notre dérivation dans **Grail** qu'une fois que tous les paramètres de la grammaire sont déterminés.

L'utilisation d'un système d'aide à la démonstration s'avère alors un bon compromis entre la généricité et l'automatisation. Parmi les assistants de preuves, on trouve le système **Coq** [Coq team, 2004, Bertot, 2004] qui est basé sur le calcul des constructions inductives : une variante de la logique intuitionniste d'ordre supérieur. Ce système d'aide à la preuve est constitué de deux composantes principales : le mode de preuves interactif et le vérificateur de types. Le premier constituant permet aux utilisateurs de construire leurs preuves interactivement grâce à l'utilisation de tactiques et procédures de décisions qui simplifient au fur et à mesure le but courant en appliquant une ou plusieurs étapes d'inférence. Quand la construction de la preuve est achevée, le vérificateur de type s'assure que cette dernière est correcte, ceci revient à vérifier que son type est convertible avec celui du but initial.

Le système **Coq** a été choisi comme méta-langage pour formaliser différentes théories logiques et ce grâce à sa richesse de types (types inductifs, types dépendants). A titre d'exemple, citons la formalisation de la logique épistémique par Pierre Lescanne [Lescanne, 2004] et la logique linéaire temporelle par Solange Coupet-Grimal [Coupet, 2003]. L'outil $\mathcal{ICHARATE}$ vient pour prolonger ces travaux en offrant une formalisation complète de la logique multimodale en **Coq**.

2 MÉTA-LINGUISTIQUE EN COQ

L'outil $\mathcal{ICHARATE}$ [Anoun, 2004, Anoun, 2005] est constitué d'un ensemble de bibliothèques **Coq** dédiées aux grammaires multimodales de Moortgat. Le noyau de l'atelier comprend la formalisation de la logique multimodale en utilisant trois systèmes déductifs qui possédent des règles de déduction différentes et présentent des avantages complémentaires [Moortgat, 1997].

$\mathcal{ICHARATE}$ contient aussi un catalogue de règles dérivées génériques, un module de calcul de la sémantique ainsi qu'une interface utilisateur qui facilite la manipulation de l'atelier pour les non spécialistes de **Coq**. Dans ce qui suit, nous présenterons une description succincte des fonctionnalités de base de cet outil. Pour plus de lisibilité, les différentes définitions et les divers théorèmes seront exposés en utilisant les notations mathématiques classiques. Toutefois, le lecteur intéressé par la formalisation en **Coq** trouvera les sources de l'atelier à l'adresse suivante : *www.labri.fr/~anoun/Icharate*.

2.1 Raisonnement sur les grammaires logiques

2.1.1 R`egles d´eriv´ees

L'ensemble des règles d'inférence de base de la logique multimodale est réduit. Pour permettre une meilleure compréhension du fonctionnement du formalisme considéré, notre atelier comprend un catalogue de règles dérivées génériques concernant différentes classes de grammaires. Ces règles dérivées constituent des schémas de dérivations, elles sont prouvées une seule fois et peuvent être utilisées autant de fois que l'on souhaite. En outre, elles facilitent aux chercheurs d'étudier la façon dont l'interaction entre différents modes permet de rendre compte de phénomènes linguistiques divers. Citons, à titre d'exemple, les deux règles dérivées d_1 et d_2 qui sont extraites de la bibliothèque d'$\mathcal{ICHARATE}$ consacrée à l'analyse des dépendances parasites (cf. Fig8).

Les règles précédentes sont génériques et universelles dans la mesure où elles font intervenir des variables formelles dont la valeur n'est pas spécifiée (e.g. les types : A, B ; les modes : i, j...). En revanche, elles sont condi-

$$\frac{MC(i,j) \in R}{(A/_j B\ ,\ C/_j B)^i \vdash A \bullet_i C/_j \Diamond_a \Box_a B}\ d_1$$

$$\frac{MC(i,j) \in R \quad L\Diamond(i,\ j,\ a) \in R}{(A, ((A\backslash_i E)/_i (C \bullet_i D), (C/_j B, D/_j B)^i)^i)^i \vdash E/_j \Diamond_a \Box_a B}\ d_2$$

FIG. 8 – Règles dérivées pour les dépendances parasites

tionnées par la présence de certaines règles structurelles dans le système considéré. Ainsi, la règle d_2 ne s'applique que pour la classe de grammaires multimodales qui supportent les deux règles de réécriture **MC(i, j)** (cf. Fig.7) et **L$\Diamond$(i, j, a)** (cf. Fig.9)[2].

$$\frac{\Gamma[(\Delta_1\ ,\ (\Delta_2, < \Delta_3 >_a)^j)^i] \vdash C}{\Gamma[((\Delta_1\ ,\ \Delta_2)^i, < \Delta_3 >_a)^j] \vdash C}\ L\Diamond(i,j,a)$$

FIG. 9 – Règle structurelle $L\Diamond$

Illustrons l'utilisation de la règle d_2 dans un exemple linguistique concret. On considère alors la phrase citée dans la section (1.3) à savoir *'The man whom I will persuade the friends of _ to vote for _ is nice'*. La partie qui nous intéresse est celle de la subordonnée relative *'whom I will persuade the friends of to vote for'* qui correspond à un modifieur de nom commun (i.e. elle joue le rôle d'un adjectif pour le nom *'man'*). On se place dans le cadre de la grammaire qui comprend les deux règles structurelles **MC(i, j)** et **L$\Diamond$(i, j, a)** ainsi que le lexique représenté dans la figure 10.

$whom$	$(n\backslash_i n)/_i(s/_j \Diamond_a \Box_a np)$
I	np
$will\ persuade$	$(np\backslash_i s)/_i(np \bullet_i inf)$
$the\ friends\ of$	$np/_j np$
$to\ vote\ for$	$inf/_j np$

FIG. 10 – Lexique 2

Il n'est pas difficile de constater qu'on peut appliquer la règle d_2 pour analyser le corps de la phrase relative comme le montre la figure 11. L'instantiation des variables formelles de la règle d_2 est immédiate (e.g. A=**np**, E=**s**, D=**inf** ...).

2.1.2 Mariage entre le raisonnement et le calcul

Coq offre un excellent environnement qui combine harmonieusement le raisonnement et le calcul. Le raisonnement est un outil plus puissant que le calcul [Dowek, 1995], toutefois, il est difficilement automatisable. Le système **Coq**, comme la majorité des assistants de preuves, vérifie le principe de Poincaré (i.e. les calculs n'ont pas besoin de preuves, ils se font de manière

$$\cfrac{\cfrac{}{(np, ((np\backslash_i s)/_i(np \bullet_i inf), (np/_j np, inf/_j np)^i)^i \vdash s/_j \Diamond_a \Box_a np}\ d2}{\cfrac{I\ will\ persuade\ the\ friends\ of\ to\ vote\ for : s/_j \Diamond_a \Box_a np}{whom\ I\ will\ persuade\ the\ friends\ of\ to\ vote\ for : n\backslash_i n}}$$

FIG. 11 – Utilisation de la règle dérivée d_2

systématique) [Barendregt, 2001]. Il est donc salutaire de pouvoir basculer, quand ceci est possible, du raisonnement au calcul pour pouvoir tirer profit de l'automatisation complète de ce dernier. Cette méthode est au coeur de la technique de *réflexion* [Alvarado, 2002] qui est fréquemment utilisée dans notre atelier. Le but de la réflexion est de pouvoir effectuer des raisonnements complexes en ayant recours au calcul. La preuve que ce raisonnement est réductible au calcul se fait une seule fois, mais elle peut être utilisée autant qu'on le souhaite pour automatiser notre raisonnement.

Nous nous servons de cette technique pour prouver formellement qu'un séquent n'est pas dérivable dans une logique donnée [3]. Pour atteindre cet objectif, il suffit de trouver une condition nécessaire définie de manière calculatoire et vérifiée par tous les séquents dérivables. Par conséquent, tous les séquents ne vérifiant pas cette condition seront forcément non démontrables.

On illustre, dans ce qui suit, l'utilisation de la technique de réflexion par le critère de non dérivabilité basé sur le calcul de polarité.

On définit la polarité d'un atome *p* dans un type syntaxique (resp. un contexte) comme étant la différence entre le nombre d'occurences positives et le nombre d'occurences négatives de cet atome dans le type (resp. le contexte) en question :

$$\begin{aligned}
pol_p(q) &= si\ (p = q)\ alors\ 1\ sinon\ 0 \\
pol_p(A/_i B) &= pol_p(A) - pol_p(B) \\
pol_p(B\backslash_i A) &= pol_p(A) - pol_p(B) \\
pol_p(A \bullet_i B) &= pol_p(A) + pol_p(B) \\
pol_p(\Diamond_j A) &= pol_p(A) \\
pol_p(\Box_j A) &= pol_p(A) \\
pol_p((\Delta_1\ ,\ \Delta_2)^i) &= pol_p(\Delta_1) + pol_p(\Delta_2) \\
pol_p(< \Delta >_j) &= pol_p(\Delta)
\end{aligned}$$

Le théorème de polarité énoncé ci dessous (cf. Fig. 12) exhibe une condition suffisante automatiquement vérifiable qui garantit la non dérivabilité d'un séquent.

$$\frac{linear(R)\ \ pol_p(\Gamma) \neq pol_p(C)}{\Gamma \nvdash C}\ Pol$$

FIG. 12 – Théorème de la polarité et réflexion

Ainsi, si on se place dans le cadre d'une grammaire dont les règles structurelles sont toutes linéaires (qui

[2] R`egle d'associativit´e restreinte ne s'appliquant que lorsque le sous contexte Δ_3 est d´ecor´e par l'op´erateur structurel '$<_a>$'

[3] Ces preuves n´egatives sont utilis´ees pour d´emontrer qu'une phrase est agrammaticale, i.e. n'est pas reconnaissable par la grammaire

conservent le nombre de ressources après la réécriture, e.g. la règle $L\diamond$(i, j, a) est linéaire alors que MC(i, j) ne l'est pas) et si on trouve un atome *p* dont la polarité dans le contexte Γ diffère de celle dans le type *C* alors il s'ensuit directement que le séquent n'est pas démontrable dans la logique sous-jacente.
Appliquons ce critère pour prouver que la phrase relative *'whom I like the friends of Houda'* est syntaxiquement incorrecte. Soit Γ un contexte structuré représentant l'expression précédente. En considérant une grammaire supportant uniquement la règle structurelle **L$\diamond$(i, j, a)** et dont le lexique est celui de la figure 10[4], on peut aisément vérifier que $pol_{np}(\Gamma)$=1 alors que $pol_{np}(n\backslash_i n)$=0, on en déduit alors que notre subordonnée relative est mal formée.

2.2 Coq et la sémantique des langues naturelles

En suivant les grammaires de **Montague** [Montague, 1974], nous utilisons le λ-calcul comme langage pivot pour accéder à l'interprétation des phrases dans des modèles. L'intérêt principal du λ-calcul étant que c'est un langage sans ambiguïté à la différence des langues naturelles. On verra dans les sections suivantes comment on peut exploiter la richesse de types de **Coq** pour exprimer, avec plus d'élégance, la sémantique lexicale des mots et raisonner ensuite sur le contenu sémantique de phrases diverses.

2.2.1 De la syntaxe `a l'interpr´etation s´emantique

Le principe de compositionnalité de **Frege** stipule que la sémantique d'une phrase est une fonction de l'arbre de dérivation syntaxique de cette dernière et la sémantique partielle des mots qui la composent. Cette fonction est systématiquement définie pour les grammaires multimodales et ce grâce à la correspondance de Curry-Howard.
L'outil $\mathcal{ICHARATE}$ contient un module qui prend en charge le calcul de la sémantique d'une phrase et ce à partir de sa dérivation syntaxique (que l'utilisateur effectue interactivement avec le système). Notre atelier offre aussi la possibilité d'interpréter cette sémantique dans différents modèles que l'utilisateur peut définir. Un modèle est défini par un domaine **D** composé d'individus (i.e. interprétation du type sémantique e) et une fonction d'interprétation **I** qui interprète les différentes constantes intervenant dans la sémantique lexicale des mots (e.g. les constantes en gras de la figure 2). Afin de se servir des quantificateurs prédéfinis de **Coq** et leurs outils de preuves, nous utilisons la sorte **Prop** des propositions logiques à la place du type **bool**.
Une phrase est sémantiquement correcte dans un modèle M(D,I) ssi la formule du calcul des prédicats représentant sa sémantique est prouvable sous les conditions du modèle en question.
Illustrons ceci par un exemple concret. On considère la phrase ambiguë *'Les filles lisent un roman'*. Suivant la manière dont l'utilisateur a effectué la dérivation syntaxique de cette phrase, on retrouve les deux sémantiques suivantes :

1. $\forall$x. FILLE(x) $\Rightarrow$ $\exists$y. ROMAN(y)$\wedge$LIRE(y,x)
2. $\exists$y. ROMAN(y) $\wedge$ $\forall$x. FILLE(x) $\Rightarrow$ LIRE(y,x)

La première interprétation de cette phrase est vraie si chaque fille du modèle lit un roman quelconque, en revanche, pour que la deuxième interprétation soit vraie, il faut qu'il existe un roman lu par toutes les filles du modèle. Testons alors la vérité de notre phrase dans le modèle **M** défini comme suit :

```
D=|houda|soumaya|amok|germinal
I(ROMAN)={germinal, amok}
I(LIRE)={(amok,houda); (germinal,soumaya)}
I(FILLE)={houda, soumaya}
```

Dans **M**, nous avons quatre individus énumérés par le domaine **D**, deux prédicats **ROMAN** et **FILLE** définissant des sous-ensembles d'individus et une relation binaire **LIRE** définie sur **D**. Il est facile de vérifier que dans ce modèle la première interprétation est correcte alors que la seconde ne l'est pas. Ces preuves sont facilitées dans $\mathcal{ICHARATE}$, ceci grâce à l'utilisation de tactiques prédéfinies en **Coq** et dédiées à la recherche de démonstration pour les théorèmes de la logique de premier ordre (e.g. tactique *firstorder*).

2.2.2 Enrichissement de la s´emantique de Montague

L'utilisation du calcul des constructions inductives pour exprimer la sémantique lexicale des mots s'avère salutaire. En effet, ceci nous permet d'enrichir la sémantique de **Montague** basée sur la logique de premier ordre en utilisant, entre autres, les spécifications fortes de **Coq** ainsi que les définitions inductives de prédicats.
Le calcul des prédicats de premier ordre ne permet pas d'exprimer élégamment la sémantique des quantificateurs indéfinis qui représentent un groupe quelconque d'individus de cardinalité bien déterminée (e.g. *un, trois, dix*). La possibilité de définir une grande variété de types abstraits de données en **Coq** nous a permis de raffiner ces expressions. Dans ce qui suit, nous proposons une définition en **Coq** de la sémantique du quantificateur *'trois'*, qu'on compare à celle de **Montague**

$Icharate$	$\lambda P.\ \lambda Q.\ \exists p : (\Sigma\ (E : fset).\|E\| = 3),$ $(\forall x,\ x \in \Pi_1(p)\ \Rightarrow\ P(x)\ \wedge\ Q(x))$
$Montague$	$\lambda P.\ \lambda Q.\ \exists\ x_1\ x_2\ x_3,\ (x_1 \neq x_2) \wedge$ $(x_2 \neq x_3) \wedge (x_1 \neq x_3) \wedge P(x_1) \wedge P(x_2)$ $\wedge P(x_3) \wedge Q(x_1) \wedge Q(x_2) \wedge Q(x_3)$

Dans cette définition, nous utilisons un type abstrait de données *fset* représentant les ensembles finis ainsi que la spécification forte (Σ)[5] pour exprimer l'existence constructive d'un élément vérifiant une certaine propriété. L'entité *p* qui est de type $(\Sigma(E : fset).|E| = 3)$ peut être vue comme une paire dont le premier élément est un ensemble fini et le second est la preuve que cet ensemble a bien la cardinalité 3. Ainsi, la sémantique de la phrase

[4] Le lexique est ´etendu en assignant au verbe transitif *'like'* le type $(np\backslash_i s)/_i np$ et au nom propre *'Houda'* la cat´egorie np

[5] Cet op´erateur est repr´esent´e par le type *sig* en **Coq**

'Trois filles lisent Amok' peut s'exprimer dans le calcul des constructions inductives de Coq comme ceci :

$$\exists p : (\Sigma(E : fset).|E| = 3),$$
$$(\forall x,\ x \in \Pi_1(p) \Rightarrow FILLE(x) \wedge LIRE(AMOK,\ x))$$

Informellement, cette proposition signifie qu'il existe un ensemble fini de cardinalité trois dont les éléments sont tous des filles qui lisent Amok.
D'autre part, nous avons eu recours aux définitions inductives afin de rassembler les différents sens d'un mot dans une seule composante sémantique. Ceci est le cas par exemple pour le mot équivoque *'bibliothèque'*.

```
Inductive Bibliotheque (ind:E):Prop:=
|sens1:endroit ind -> contenir_livres ind
       -> Bibliotheque ind
|sens2:collection_prgms ind->
        Bibliotheque ind.
```

La définition ci dessus précise qu'une *'bibliothèque* peut être soit un endroit contenant des livres (énoncé du constructeur sens1) soit une collection de programmes (énoncé du constructeur sens2).

2.2.3 S´emantique & Raisonnement

Le fait d'exprimer la sémantique des phrases dans le calcul des constructions inductives nous permet de raisonner sur leur contenu. A titre d'exemple, on est capable de prouver que les deux phrases *'Le livre que lit Houda s'intitule Amok'* et *'Houda lit le livre qui s'intitule Amok'* sont sémantiquement équivalentes (i.e. leur interprétation est la même dans tous les modèles) même si elles sont totalement différentes au niveau syntaxique.
Le raisonnement sur le contenu sémantique de phrases nous permet de prouver formellement différents syllogismes qui sont valides dans tous les modèles tel le syllogisme illustré dans la figure 13[6].

$$\frac{M \models Tous\ les\ hommes\ sont\ mortels \quad M \models Socrate\ est\ un\ homme}{M \models Socrate\ est\ mortel}$$

FIG. 13 – Exemple de syllogisme

Finalement, nous utilisons l'assistant **Coq** pour tester la validité de certaines phrases méta-mathématiques (i.e. où on utilise une langue naturelle pour décrire des propriétés mathématiques). Ainsi, on peut démontrer que la phrase *'Trois plus quatre est égal à sept'* est valide à la différence de *'Zéro est strictement positif'* qui ne l'est pas.

2.3 Interface utilisateur

2.3.1 Banque de tactiques

$\mathcal{ICHARAT\mathcal{E}}$ est muni d'une banque de tactiques spécialisées définies dans le langage **Ltac** qui est fourni avec le système **Coq** [Delahaye, 2001]. Ce dernier langage est assez riche pour garantir la définition de tactiques efficaces récursives ou non sans avoir à passer par la programmation en **OCaml**.
Le but de cette banque de tactiques est de mettre à la disposition des utilisateurs des outils de semi-automatisation, des tactiques de réfutation et des procédures de décision qui faciliteront leurs preuves en logique multimodale. Ces tactiques peuvent intervenir aussi bien dans les preuves de nouvelles règles dérivées génériques que dans les dérivations syntaxiques particulières effectuées dans une grammaire donnée. Elles permettent de simplifier le but courant en appliquant un ensemble de règles (de base ou dérivées), ce qui engendre une liste de sous-buts potentiellement plus simples à résoudre. Les tactiques sont programmées de façon à automatiser les preuves de certains sous-buts triviaux qui peuvent être engendrés par l'application des règles dérivées (e.g. vérification de la linéarité des règles structurelles de la grammaire...).
Les tactiques de réfutation, quant à elles, essaient de prouver la non dérivabilité d'un séquent en appliquant, un par un, les différents critères de non démontrabilité prouvés dans l'atelier. Dans chacun des cas, la vérification des contraintes associées se fait de manière automatique.

2.3.2 PCoq et proof-by-pointing

Afin de définir une syntaxe concrète lisible et compréhensible par les utilisateurs, nous avons eu recours à l'utilisation de l'interface graphique **PCoq** [Amerkad, 2001]. **PCoq** est une interface graphique conviviale dédiée à **Coq**, elle facilite la définition de différentes règles d'affichage et offre la possibilité d'utiliser les notations mathématiques classiques (e.g. $\forall$, $\exists$, $\vdash$, ...).
En plus, **PCoq** assure une grande interactivité avec les utilisateurs notamment grâce à la technique de *proof-by-pointing* [Bertot, 1994]. Cette technique permet de simplifier le but à résoudre et ce par l'intermédiaire de simples clics de souris. En analysant la structure du but courant et la position des clics de la souris, le système engendre automatiquement des commandes complexes qu'il renvoie à **Coq**, ceci permet de progresser dans la preuve sans avoir à connaître la syntaxe exacte des tactiques.
L'algorithme de proof-by-pointing pour la logique intuitionniste a été implémenté en **OCaml** par l'équipe *Lemme*[7] et ce de manière récursive et facilement extensible. Nous avons exploité ces atouts pour étendre les règles de cette technique à la logique multimodale. Plusieurs règles ont été définies dans notre atelier notamment pour introduire les hypothèses dans le contexte, ainsi que pour se focaliser sur un sous-contexte afin d'appliquer une règle de réécriture. La figure 14 montre un exemple d'une règle dérivée de la logique multimodale dont la preuve se fait par un simple clic de souris sur le type **A** le plus imbriqué de la conclusion (encadré en gris). Les étapes de la dérivation engendrées automatiquement sont illustrées dans cette même figure.

[6] Pour une phrase *sent* non ambigue, la notation $\mathbf{M} \models$ *sent* exprime le fait que cette phrase est vraie dans le mod`ele **M**

[7] Inria Sophia Antipolis

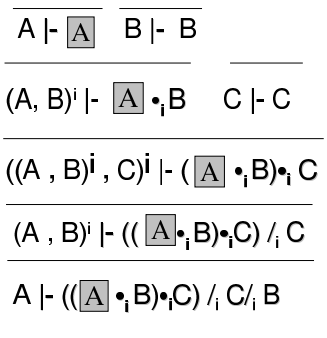

FIG. 14 – Proof by pointing

CONCLUSION ET PERSPECTIVES

L'atelier logique $\mathcal{ICHARATE}$ que nous avons présenté dans cet article est un projet en cours de réalisation. Notre objectif est de mettre à la disposition des chercheurs un outil convivial qui leur permet d'étudier et d'appréhender des formalismes logiques complexes dédiés au traitement des langues naturelles dont la logique multimodale.

Cet outil comprend un catalogue de règles dérivées qui s'appliquent pour différentes classes de grammaires et qui facilitent l'analyse de divers phénomènes linguistiques ainsi qu'un module de calcul de la sémantique doté d'outils variés facilitant l'interprétation de cette sémantique dans des modèles.

En outre, quelques tests concernant l'enrichissement de la sémantique de **Montague** ont été réalisés. Nous envisageons d'explorer cette voie de recherche dans le but de remédier contre les faiblesses de cette sémantique simpliste (e.g. problème de résolution des anaphores) en s'inspirant des travaux d'Aarne Ranta [Ranta, 2004].

Dans une autre direction, un autre formalisme logique conçu par Alain Lecomte [Lecomte, 2004] est en cours de formalisation en **Coq**. Son intégration à l'atelier $\mathcal{ICHARATE}$ permettra de tester sa pertinence, faciliter l'étude de ses propriétés logiques et combler ses éventuelles lacunes.

Finalement, plusieurs efforts restent à faire pour améliorer l'interface utilisateur de notre atelier, notamment pour faciliter la saisie des données (e.g. règles structurelles, lexiques,...).

REMERCIEMENTS

Je tiens à remercier Pierre Castéran, Paul Gloess et Alain Lecomte pour leur encouragement, leurs remarques judicieuses et discussions fructueuses. Je remercie aussi de tout coeur, mes chers parents et ma soeur Soumaya pour leur soutien moral permanent qui est toujours aussi fort malgré la grande distance qui nous sépare.

BIBLIOGRAPHIE

[Alvarado, 2002] Alvarado C., *Réflexion pour la réécriture dans le calcul des constructions inductives*. PhD thesis, Université de Paris Sud, Orsay.

[Amerkad, 2001] Amerkad A., Bertot Y., Pottier L. et Rideau L., Mathematics and proof presentation in Pcoq. Rapport technique, INRIA, Sophia Antipolis.

[Anoun, 2005] Anoun H., Une bibliothèque Coq pour le traitement des langues naturelles. *JFLA*, Obernai.

[Anoun, 2004] Anoun H., Castéran P. et Moot R., Proof automation for type-logical grammars. Rapport technique, Notes pour le cours d'ESSLLI, Nancy.

[Barendregt, 2001] Barendregt H. et Geuvers H., Proof assistants using dependent type systems. *Handbook of Automated Reasoning*. Elsevier.

[Bertot, 2004] Bertot Y. et Castéran P., *Interactive Theorem Proving and Program Development. Coq'Art : The Calculus of Inductive Constructions*. Springer Verlag.

[Bertot, 1994] Bertot Y., Kahn G. et Théry L., Proof by Pointing. *Symposium on Theoretical Aspects of Computer Software*, Japon.

[Chomsky, 1957] Chomsky N., *Syntactic structures*. Mouton & Co.

[Coq team, 2004] Coq team, The Coq proof assistant, reference manual.

[Coupet, 2003] Coupet S., An axiomatization of linear temporal logic in the calculus of inductive constructions. *Journal of Logic and Computation*, 13(6) :801–813.

[Delahaye, 2001] Delahaye D., *Conception de langages pour décrire les preuves et les automatisations dans les outils d'aide à la preuve*. PhD thesis, Université de Paris VI, Pierre et Marie Curie.

[Dowek, 1995] Dowek G., *La logique*. Flammarion.

[GAMUT, 1991] GAMUT L., *Language and Meaning, Vol 2 : Intentional Logic and Logical Grammar*. University of Chicago Press.

[Girard, 1987] Girard J.-Y., Linear Logic. *Theoretical Computer Science*, 50 :1–102.

[Huet, 1997] Huet G., The Zipper. *Journal Of Functional Programming*.

[Lambek, 1961] Lambek J., On the calculus of syntactic types. *Structure of Language and its Mathematical Aspects*.

[Lecomte, 2004] Lecomte A., Derivations as Proofs, a Logical Approach to Minimalism.

[Lescanne, 2004] Lescanne P., Mechanizing epistemic logic with coq. Rapport technique, ENS Lyon.

[Montague, 1974] Montague R., The proper treatment of quantification in ordinary English. *Formal Philosophy. Selected Papers of Richard Montague*. Yale University Press.

[Moortgat, 1997] Moortgat M., Categorial Type Logics. In Van Benthem, J. & ter Meulen A., editor, *Handbook of Logic and Language*, chapter 2. Elsevier/MIT Press.

[Moot, 2002] Moot R., *Proof Nets for Linguistic Analysis*. PhD thesis, Utrecht University.

[Ranta, 2004] Ranta A., Computational Semantics in Type Theory. *Mathématiques et sciences humaines*.

[Ross, 1967] Ross J.-R., *Constraints on Variables in Syntax*. PhD thesis, MIT.

[Steedman, 1996] Steedman M., *Surface Structure and Interpretation*. The MIT Press.