Rigid Lambek grammars are not learnable from strings

Annie Foret and **Yannick Le Nir**
IRISA Rennes, FRANCE
e-mail: foret@irisa.fr and ylenir@irisa.fr

Abstract

This paper is concerned with learning categorial grammars in Gold's model (Gold, 1967). Recently, learning algorithms in this model have been proposed for some particular classes of classical categorial grammars (Kanazawa, 1998).

We show that in contrast to classical categorial grammars, rigid and k-valued Lambek grammars are not learnable from strings. This result holds for variants of Lambek calculus ; our proof consists in the construction of limit points in each class. Such a result aims at clarifying the possible directions for future learning algorithms.

1 Introduction

Categorial grammars have been studied in the field of natural language processing, classical (or basic) categorial grammars were introduced in (Bar-Hillel, 1953) ; here we focus on Lambek categorial grammars (Lambek, 1958) to which linear logic introduced by Girard (Girard, 1995) is closely connected. These grammars are lexicalized grammars that assign types (or categories) to the lexicon; they are called *k-valued*, when each symbol in the lexicon is assigned to at most k types; they are also called *rigid* when 1-valued. Such k-valued grammars are of particular interest in recent works on learnability (Kanazawa, 1998) (Nicolas, 1999). The issue of extending Kanazawa's work to other categorial grammars has been raised and has become an active area of research. In this context, it is important to acquire a good understanding of the properties of the class of grammars in question.

In this paper we consider the following problem, is the class of rigid Lambek grammars learnable from strings. Learning (in the sense of Gold (Gold, 1967)) in our context is a symbolic issue that may be described as follows. Let $\mathcal{G}$ be a class of grammars, that we wish to learn from examples. The issue is to define an algorithm, that when applied to a finite set of sentences, yields a grammar in the class that generates the examples; the algorithm is also required to converge. Formally, let $\mathcal{L}(G)$ denote the language associated with grammar G, and let V be a given alphabet, a learning algorithm is a function ϕ from finite sets of words in V^* to $\mathcal{G}$, such that for $G \in \mathcal{G}$ with $\mathcal{L}(G) = (e_i)_{i \in N}$ there exists a grammar $G' \in \mathcal{G}$ and there exists $n_0 \in N$ such that : $\forall n > n_0 \ \phi(\{e_1, \ldots, e_n\}) = G' \in \mathcal{G}$ with $\mathcal{L}(G') = \mathcal{L}(G)$.

One good reason to use categorial grammars in a learning perspective is that they are fully lexicalized : the rules are already known, only types assigned to words have to be derived from examples.

The paper is organized as follows. Section 2 addresses background definition and known results. We then proceed from one version of Lambek calculus to the other. Section 3 gives the initial construction and proof for Lambek calculus allowing empty sequences. Section 4 adresses the construction for Lambek calculus without empty sequence including products. Section 5 concludes.

2 Background

2.1 Categorial grammars

In this section, we introduce basic definitions concerning categorial grammars. The interested reader may also consult (Casadio, 1988; Retoré, 2000; Buszkowski, 1997; Moortgat, 1997) for an introduction or for further details.

Let Σ be a fixed alphabet.

Types. *Types* are constructed from Pr (set of *primitive types*) and three binary connectives

$/$, $\backslash$ and $\bullet$ for products. Tp denotes the set of types. Pr contains a *distinguished type*, written S, also called the *principal type*.

Categorial grammar. A *categorial grammar* over Σ is a finite relation G between Σ and Tp. If $< c, A > \in G$, we say that G *assigns* A to c, and we write $G : c \mapsto A$.

We give a formulation of Lambek calculus, written L, including products consisting in introduction rules on the left and on the right of a sequent. For Lambek calculus without products, one simply drops the rules for $\bullet$.

Lambek Derivation $\vdash_L$. The relation $\vdash_L$ is the smallest relation $\vdash$ between Tp^+ and Tp, such that for all $\Gamma, \Gamma' \in Tp^+, \Delta, \Delta' \in Tp^*$ and for all $A, B \in Tp$:

$$A \vdash A$$

$$\frac{A, \Gamma \vdash B}{\Gamma \vdash A \backslash B} \backslash r \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B / A} / r$$

$$\frac{\Gamma \vdash A \quad \Delta, B, \Delta' \vdash C}{\Delta, \Gamma, A \backslash B, \Delta' \vdash C} \backslash l \qquad \frac{\Gamma \vdash A \quad \Delta, B, \Delta' \vdash C}{\Delta, B / A, \Gamma, \Delta' \vdash C} / l$$

$$\frac{\Delta, A, B, \Delta' \vdash C}{\Delta, (A \bullet B), \Delta' \vdash C} \bullet l \qquad \frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash (A \bullet B)} \bullet r$$

When we replace Tp^+ by Tp^* in $\Gamma \in Tp^+$ in the definition above, we get another version of Lambek calculus, without the non-empty left hand-side requirement, which we refer to as $L_\emptyset$ with derivation relation $\vdash_{L_\emptyset}$.

Note. We recall that the cut rule is satisfied by $\vdash_L$ and $\vdash_{L_\emptyset}$. Note also that $\Gamma \vdash_L C$ implies $\Gamma \vdash_{L_\emptyset} C$

Language. Let G be a categorial grammar over Σ. G *generates* a string $c_1 \ldots c_n \in \Sigma^+$ iff there are types $A_1, \ldots, A_n \in Tp$ such that : $G : c_i \mapsto A_i$ $(1 \leq i \leq n)$ and $A_1, \ldots, A_n \vdash_L S$. The *language of* G, written $\mathcal{L}_L(G)$ is the set of strings generated by G. We define similarly $\mathcal{L}_{L_\emptyset}(G)$ replacing $\vdash_L$ with $\vdash_{L_\emptyset}$.

Notation. In some sections, we may write simply $\vdash$ instead of $\vdash_L$ or $\vdash_{L_\emptyset}$. We may simply write $\mathcal{L}(\mathcal{G})$ accordingly.

Rigid and k-valued grammars. Categorial grammars that assign at most k types to each symbol in the alphabet are called *k-valued grammars*; 1-valued grammars are also called *rigid* grammars.

Example 1 *Let* $\Sigma_1 = \{John, Mary, likes\}$ *and let* $Pr = \{S, N\}$ *for sentences and nouns respectively. Let* $G_1 = \{John \mapsto N,\ Mary \mapsto N,\ likes \mapsto N \backslash (S / N)\}$. *We get* $(John\ likes\ Mary) \in \mathcal{L}_L(G_1)$ *since* $(N,\ N \backslash (S / N),\ N \vdash_L S)$.

G_1 is a rigid (or 1-valued) grammar.

2.2 Some useful models

For ease of proof, in next section we use models of L (or $L_\emptyset$) that we now recall : powerset residuated semi-groups (or monoids), a special case of residuated semi-groups (see (Buszkowski, 1997) for details).

Powerset residuated semi-groups and monoids. Let $(M, .)$ be a semi-group (. is associative). Let $\mathcal{P}(M)$ denote the powerset of M. A *powerset residuated semi-group* over $(M, .)$ is the structure $(\mathcal{P}(M), \circ, \Rightarrow, \Leftarrow, \subseteq)$ such that for $X, Y \subseteq M$:

$X \circ Y = \{x.y : x \in X, y \in Y\}$
$X \Rightarrow Y = \{y \in M : (\forall x \in X) x.y \in Y\}$
$Y \Leftarrow X = \{y \in M : (\forall x \in X) y.x \in Y\}$

If $(M, \circ)$ is a monoid (. is associative, I is a unit that is : $\forall x \in M : I.x = x.I = x$), then the above structure is a *powerset residuated monoid* (it has $\{I\}$ as unit).

Interpretation. Given a powerset residuated semi-group $(\mathcal{P}(M), \circ, \Rightarrow, \Leftarrow, \subseteq)$, an *interpretation* is a map from primitive types p to elements $[[p]]$ in $\mathcal{P}(M)$ that is extended to types and sequences in the natural way :

$[[C_1 \backslash C_2]] = [[C_1]] \Rightarrow [[C_2]]$
$[[C_1 / C_2]] = [[C_1]] \Leftarrow [[C_2]]$
$[[C_1 \bullet C_2]] = [[C_1]] \circ [[C_2]]$
$[[C_1, C_2, \ldots, C_n]] = [[C_1]] \circ [[C_2]]. \ \ldots \ \circ [[C_n]]$

The following known property states that such structures are models for L : if $\Gamma \vdash_L C$ then $[[\Gamma]] \subseteq [[C]]$

If $(M, .)$ is a monoid with an identity I, we add $[[\Lambda]] = \{I\}$ for the empty sequence Λ and get a similar property for $L_\emptyset$: if $\Gamma \vdash_{L_\emptyset} C$ then $[[\Gamma]] \subseteq [[C]]$

2.3 Learning and limit points

We now recall some useful definitions and known properties on learning.

Limit points. A class $\mathcal{CL}$ of languages has *a limit point* iff there exists an infinite sequence $< L_n >_{n \in N}$ of languages in $\mathcal{CL}$ and a language $L \in \mathcal{CL}$ such that : $L_0 \subsetneq L_1 \ldots \subsetneq ... \subsetneq L_n \subsetneq \ldots$ and $L = \bigcup_{n \in N} L_n$ (L is *a limit point* of $\mathcal{CL}$).

Limit points imply unlearnability. The following property is important for our purpose. If the languages of the grammars in a class $\mathcal{G}$ have a limit point then the class $\mathcal{G}$ is *unlearnable*. [1]

3 Rigid limit points for $L_\emptyset$

3.1 Construction overview

Definition. We define the following grammars where p and q are primitive types :
$G_{\langle 1,n \rangle} = \{a \rightarrow\ p \,/\, p;\ \ b \rightarrow\ q \,/\, q;\ \ c \rightarrow\ D_{\langle 1,n \rangle}\}$
where $D_{\langle 1,0 \rangle} = S$
and $D_{\langle 1,n \rangle} = (D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p)) \,/\, (q \,/\, q)$
$G_{\langle 1,* \rangle} = \{a \rightarrow\ p \,/\, p;\ b \rightarrow\ p \,/\, p;\ c \rightarrow\ S \,/\, (p \,/\, p)\}$

Language. We get (see proof) $\mathcal{L}(G_{\langle 1,n \rangle}) = c(b^*a^*)^n$ and $\mathcal{L}(G_{\langle 1,* \rangle}) = c(b^*a^*)^* = c\{a,b\}^*$.

Notation. Let $\tau_{\langle 1,n \rangle}$ (and $\tau_{\langle 1,* \rangle}$) denote the type assignment by $G_{\langle 1,n \rangle}$ (by $G_{\langle 1,* \rangle}$ respectively) on $\{a,b,c\}$ extended on $\{a,b,c\}^*$ to sequences of types in the natural way; we write $\tau = \tau_{\langle 1,n \rangle}$ on $\{a,b\}^*$ (independant of $n \geq 0$).

Key points. We use two main key ideas : tautologies of the Lambek calculus allowing empty sequences that ensure one way of type-derivability ($D_{\langle 1,n \rangle} \vdash D_{\langle 1,n-1 \rangle}$) ; an alternation of two such tautologies that are unrelated (non-interderivable : $\tau(a), \tau(b) \not\vdash \tau(a)$ or $\tau(a), \tau(b) \not\vdash \tau(b)$ although we have $\tau(a), \tau(a) \vdash \tau(a)$), this alternation prevents derivabilities in the other direction ($D_{\langle 1,n-1 \rangle} \not\vdash D_{\langle 1,n \rangle}$). We thus provide a strictly infinite chain of types. Note that n denotes a bound of these alternations.

[1] This implies that the class has infinite elasticity. A class $\mathcal{CL}$ of languages has *infinite elasticity* iff $\exists < e_i >_{i \in N}$ sentences $\exists < L_i >_{i \in N}$ languages in $\mathcal{CL}$ $\forall i \in N : e_i \notin L_i$ and $\{e_1, \ldots, e_n\} \subseteq L_{n+1}$ (see (Kanazawa, 1998) for this notion and a use of it).

3.2 Corollaries

For the class of rigid $L_\emptyset$-grammars.
- This yields a strictly increasing chain of language of rigid grammars.
- This shows that the class of rigid grammars has **infinite elasticity** (cf (Kanazawa, 1998) for details).
- This class also **has a limit point** as follows $c\{a,b\}^*$ which entails that this class is **not learnable from strings.**

Other restricted subclasses

- The same results hold if we restrict to a bounded order, where the order $o(A)$ is :
$o(p) = 0$ when p is a primitive type
$o(C_1 \setminus C_2) = max(o(C_1) + 1, o(C_2))$
$o(C_2 \,/\, C_1) = o(C_1 \setminus C_2)$;
- this also holds for unidirectional grammars (we do not use $\setminus$).

3.3 Details of proofs

Our proof is based both on a syntactic reasoning on derivations, and on models.

Proposition 1 (Language description)
$\mathcal{L}(G_{\langle 1,n \rangle}) = c(b^*a^*)^n$ and $\mathcal{L}(G_{\langle 1,* \rangle}) = c\{a,b\}^*$.

proof of $c(b^*a^*)^n \subseteq \mathcal{L}(G_{\langle 1,n \rangle})$
For $n = 0$ this is an axiom $\tau_{\langle 1,0 \rangle}(c) = S \vdash S$.
Suppose $n > 0$ and $w' = c.w \in \mathcal{L}(G_{\langle 1,n-1 \rangle})$,
- we first show that $c.b.a.w \in \mathcal{L}(G_{\langle 1,n \rangle})$:

$$\dfrac{\dfrac{\dfrac{\vdots}{D_{\langle 1,n-1 \rangle}, \tau(w) \vdash S} \quad p \,/\, p \vdash p \,/\, p}{D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p), (p \,/\, p), \tau(w) \vdash S} \,/\, l \quad q \,/\, q \vdash q \,/\, q}{\underbrace{(D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p)) \,/\, (q \,/\, q)}_{=D_{\langle 1,n \rangle} = \tau_{\langle 1,n \rangle}(c)}, \underbrace{(q \,/\, q)}_{=\tau(b)}, \underbrace{(p \,/\, p)}_{=\tau(a)}, \tau(w) \vdash S} \,/\, l$$

- we easily get $c.w \in \mathcal{L}(G_{\langle 1,n \rangle})$ from $D_{\langle 1,n \rangle} \vdash D_{\langle 1,n-1 \rangle}$ in $L_\emptyset$ for $n > 0$; (more generally $C_1 \,/\, (C_2 \,/\, C_2) \vdash C_1$ in $L_\emptyset$);
- we also get $c.a.w \in \mathcal{L}(G_{\langle 1,n \rangle})$
from $D_{\langle 1,n \rangle}, p \,/\, p \vdash D_{\langle 1,n-1 \rangle}$
(since $(D_{\langle 1,n \rangle} \vdash D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p)$
and $D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p), p \,/\, p \vdash D_{\langle 1,n-1 \rangle})$
- we then get $c.b.w \in \mathcal{L}(G_{\langle 1,n \rangle})$ since
$D_{\langle 1,n \rangle}, q \,/\, q = (D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p)) \,/\, (q \,/\, q), q \,/\, q$
$\vdash D_{\langle 1,n-1 \rangle} \,/\, (p \,/\, p) \vdash D_{\langle 1,n-1 \rangle}$;
- finally, this is extended to repetitions of each letter a or b separately since $\tau(a), \tau(a) \vdash \tau(a)$

and $\tau(b), \tau(b) \vdash \tau(b)$ (if we replace each occurrence of a with a repetition of a or each occurrence of b with a repetition of b we still get an element of the language of $G_{\langle 1,n\rangle}$).

proof of $\mathcal{L}(G_{\langle 1,n\rangle}) \subseteq c(b^*a^*)^n$ (main part)
We consider the standard linguistic interpretation (Buszkowski, 1997) : the powerset residuated monoid $(\mathcal{P}(V^*), \circ, \Rightarrow, \Leftarrow, \subseteq)$ over the free monoid $(V^*, .)$ where . is the concatenation operation and V^* is the set of strings over the alphabet $V = \{a, b, c\}$.

Let us fix n (arbitrary), we define an interpretation as follows : $[[S]] = c(b^*a^*)^n$, $[[p]] = a^*$, $[[q]] = b^*$. Let us suppose $\tau_{\langle 1,n\rangle}(w) \vdash S$. By models, we have $[[\tau_{\langle 1,n\rangle}(w)]] \subseteq [[S]]$. We first remark that $[[p \,/\, p]] = a^*$ and $[[q \,/\, q]] = b^*$ (since $[[p \,/\, p]] = \{z \in V^* : \forall x \in [[p]], z.x \in [[p]]\} = \{z \in V^* : \forall x \in a^*, z.x \in a^*\}$).

We now show by induction on i that :
$\forall i (0 \leq i \leq n)$: $[[D_{\langle 1,i\rangle}]] = c(b^*a^*)^{n-i}$
- case $i = 0 \leq n$ holds since $[[D_{\langle 1,0\rangle}]] = [[S]] = c(b^*a^*)^n$
- case $(0 < i \leq n)$:
$[[(D_{\langle 1,i-1\rangle} \,/\, (p \,/\, p))]]$
$= \{z \in V^* : \forall x \in [[p \,/\, p]], z.x \in [[D_{\langle 1,i-1\rangle}]]\}$
$=_{ind.} \{z \in V^* : \forall x \in a^*, z.x \in c(b^*a^*)^{(n-(i-1))}\}$
$= [[D_{\langle 1,i-1\rangle}]]$ $(= c(b^*a^*)^{(n-(i-1))})$
$[[D_{\langle 1,i\rangle}]] = [[(D_{\langle 1,i-1\rangle} \,/\, (p \,/\, p)) \,/\, (q \,/\, q)]]$
$= \{z \in V^* : \forall y \in [[q \,/\, q]],$
$z.y \in [[D_{\langle 1,i-1\rangle} \,/\, (p \,/\, p)]]\}$
$= \{z \in V^* : \forall y \in b^*, z.y \in c(b^*a^*)^{(n+1-i)}\}$
$= \{z \in V^* : \forall y \in b^*, z.y \in c(b^*a^*)^{(n-i)}.b^*.a^*\}$
(from above)
$= c(b^*a^*)^{(n-i)}$ (as desired)

We have thus shown that $[[D_{\langle 1,n\rangle}]] = c$. Therefore if $[[\tau_{\langle 1,n\rangle}(w)]] \subseteq [[S]]$ this also means that $w = cw'$ with $w' \in \{a, b\}^*$; we get $c.[[\tau_{\langle 1,n\rangle}(w')]] \subseteq c(b^*a^*)^n = [[S]]$, that is $[[\tau_{\langle 1,n\rangle}(w')]] \subseteq (b^*a^*)^n$ that corresponds to $w' \in (b^*a^*)^n$ as well (since $[[\tau_{\langle 1,n\rangle}(a)]] = [[p \,/\, p]] = a^*$ and $[[\tau_{\langle 1,n\rangle}(b)]] = [[q \,/\, q]] = b^*$).

proof of $c\{a, b\}^* \subseteq \mathcal{L}(G_{\langle 1,*\rangle})$
- We have $c \in \mathcal{L}(G_{\langle 1,*\rangle})$ since $S \,/\, (p \,/\, p) \vdash_{L_\emptyset} S$.
- We now get $ca \in \mathcal{L}(G_{\langle 1,*\rangle})$, since :

$$\frac{p \,/\, p \vdash p \,/\, p \quad S \vdash S}{S \,/\, (p \,/\, p), p \,/\, p \vdash S}$$

The other cases are straightforward since $p \,/\, p, p \,/\, p \vdash p \,/\, p$.

proof of $\mathcal{L}(G_{\langle 1,*\rangle}) \subseteq c\{a, b\}^*$
We consider the powerset residuated monoid $(\mathcal{P}(V^*), \circ, \Rightarrow, \Leftarrow, \subseteq)$ as above but with the following (similar) interpretation :
$[[S]] = c\{a, b\}^*$, $[[p]] = a^*$, $([[q]] = b^*)$.
Let us suppose $\tau_{\langle 1,*\rangle}(w) \vdash S$.
By models, we have $[[\tau_{\langle 1,*\rangle}(w)]] \subseteq [[S]]$.
We have : $[[p \,/\, p]] = a^*$
We here get : $[[S \,/\, (p \,/\, p)]] = c.\{a, b\}^*$ $(= \{z \in V^* : \forall x \in [[p \,/\, p]], z.x \in [[S]]\} = \{z \in V^* : \forall x \in a^*, z.x \in c.\{a, b\}^*\})$
Therefore if $[[\tau_{\langle 1,*\rangle}(w)]] \subseteq [[S]] = c\{a, b\}^*$, this also means that $w = cw'$ with $w' \in \{a, b\}^*$ as desired ■

4 Rigid limit points for L

Key points: We have $D_{\langle 1,n\rangle} \vdash_{L_\emptyset} D_{\langle 1,n-1\rangle}$ but $D_{\langle 1,n\rangle} \not\vdash_L D_{\langle 1,n-1\rangle}$. We then transform the type of c from $\tau_{\langle 1,n\rangle}$ and construct a limit point for L (with products).

Construction for L
We define the following types and assignments $\tau_{\langle 2,n\rangle}$, where $A = p \,/\, p$, $B = q \,/\, q$ and p, q are primitive types :
$a \to A$; $b \to B$; $c \to D_{\langle 2,n\rangle}$
where $D_{\langle 2,0\rangle} = S$
and $D_{\langle 2,n\rangle} = (((D_{\langle 2,n-1\rangle} \,/\, A) \bullet A) \,/\, B) \bullet B$ if $n > 0$
Let $G_{\langle 2,n\rangle}$ denote the grammar defined by $\tau_{\langle 2,n\rangle}$ with alphabet $\{a, b, c\}$.
Let $G_{\langle 2,*\rangle}$ denote the grammar, with type assignment $\tau_{\langle 2,*\rangle}$ defined by :
$G_{\langle 2,*\rangle} = \{a \to A;\ b \to A;\ c \to (S \,/\, A) \bullet A\}$
We get (see proof) $\mathcal{L}(G_{\langle 2,n\rangle}) = c(b^*a^*)^n$ and $\mathcal{L}(G_{\langle 2,*\rangle}) = c\{a, b\}^*$.

Proposition 2 (Language description)
$\mathcal{L}(G_{\langle 2,n\rangle}) = c(b^*a^*)^n$ and $\mathcal{L}(G_{\langle 2,*\rangle}) = c\{a, b\}^*$.

proof of $c(b^*a^*)^n \subseteq \mathcal{L}(G_{\langle 2,n\rangle})$
For $n = 0$, $\tau_{\langle 2,0\rangle}(c) = S \vdash S$.
Suppose $n > 0$ and $w' = c.w \in \mathcal{L}(G_{\langle 2,n-1\rangle})$
- we first show that $c.b.a.w \in \mathcal{L}(G_{\langle 2,n\rangle})$

$$\dfrac{\dfrac{\dfrac{\vdots}{D_{\langle 2,n-1\rangle}, \tau_{\langle 2,n\rangle}(w) \vdash S} \quad A, A \vdash A}{(D_{\langle 2,n-1\rangle} \,/\, A) \bullet A, A, \tau_{\langle 2,n\rangle}(w) \vdash S} \quad B, B, \vdash B}{\underbrace{(((D_{\langle 2,n-1\rangle} \,/\, A) \bullet A) \,/\, B) \bullet B}_{=D_{\langle 2,n\rangle}=\tau_{\langle 2,n\rangle}(c)}, \underbrace{B}_{=\tau(b)}, \underbrace{A}_{=\tau(a)}, \tau_{\langle 2,n\rangle}(w) \vdash S}$$

- we easily get $c.w \in \mathcal{L}(G_{\langle 2,n\rangle})$ since $D_{\langle 2,n\rangle} \vdash D_{\langle 2,n-1\rangle}$ in L for $n > 0$.
- we also get $c.a.w \in \mathcal{L}(G_{\langle 2,n\rangle})$ from $D_{\langle 2,n\rangle}, A \vdash D_{\langle 2,n-1\rangle}$
(since $D_{\langle 2,n\rangle} \vdash (D_{\langle 2,n-1\rangle} \,/\, A) \bullet A$
and $(D_{\langle 2,n-1\rangle} \,/\, A) \bullet A, A \vdash D_{\langle 2,n-1\rangle}$)
- we then get $c.b.w \in \mathcal{L}(G_{\langle 2,n\rangle})$ since
$D_{\langle 2,n\rangle}, B = (((D_{\langle 2,n-1\rangle} \,/\, A) \bullet A) \,/\, B) \bullet B, B$
$\vdash (D_{\langle 2,n-1\rangle} \,/\, A) \bullet A \vdash D_{\langle 2,n-1\rangle}$;
- finally, this is extended to repetitions of each letter a or b separately since $\tau(a), \tau(a) \vdash \tau(a)$ and $\tau(b), \tau(b) \vdash \tau(b)$

proof of $\mathcal{L}(G_{\langle 2,n\rangle}) \subseteq c(b^*a^*)^n$

We first show that $D_{\langle 1,n\rangle} \vdash_{L_\emptyset} D_{\langle 2,n\rangle}$ by induction on n :
- case $n = 0$ is obvious since $D_{\langle 1,0\rangle} = D_{\langle 2,0\rangle} = S$
- case $n > 0$: we recall that $A = p \,/\, p$, $B = q \,/\, q$ are tautologies of $L_\emptyset$;
by induction : $D_{\langle 1,n-1\rangle} \vdash_{L_\emptyset} D_{\langle 2,n-1\rangle}$
we then get in $L_\emptyset$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{D_{\langle 1,n-1\rangle} \vdash D_{\langle 2,n-1\rangle}} \quad A \vdash A}{D_{\langle 1,n-1\rangle} \,/\, A, A \vdash D_{\langle 2,n-1\rangle}}}{D_{\langle 1,n-1\rangle} \,/\, A \vdash D_{\langle 2,n-1\rangle} \,/\, A} \quad B \vdash B}}{(D_{\langle 1,n-1\rangle} \,/\, A) \,/\, B, B \vdash D_{\langle 2,n-1\rangle} \,/\, A} \quad \vdots \vdash A}{(D_{\langle 1,n-1\rangle} \,/\, A) \,/\, B, B \vdash (D_{\langle 2,n-1\rangle} \,/\, A) \bullet A}}{(D_{\langle 1,n-1\rangle} \,/\, A) \,/\, B \vdash ((D_{\langle 2,n-1\rangle} \,/\, A) \bullet A) \,/\, B} \quad \vdots \vdash B}{(D_{\langle 1,n-1\rangle} \,/\, A) \,/\, B \vdash ((D_{\langle 2,n-1\rangle} \,/\, A) \bullet A) \,/\, B) \bullet B}$$

We may now end the proof : suppose that $\tau_{\langle 2,n\rangle}(w) \vdash_L S$, we get $\tau_{\langle 1,n\rangle}(w) \vdash_{L_\emptyset} S$ from previous property, therefore $w \in c(b^*a^*)^n$ from proposition 1 as desired.

proof of $c\{a,b\}^* \subseteq \mathcal{L}(G_{\langle 2,*\rangle})$
- We have $c \in \mathcal{L}(G_{\langle 2,*\rangle})$ since $(S \,/\, A) \bullet A \vdash_L S$.
- We also get $ca \in \mathcal{L}(G_{\langle 2,*\rangle})$, and similarly for ca^* or $c\{a,b\}^*$ since $A, A \vdash_L A$ (where $A = p \,/\, p$).

proof of $\mathcal{L}(G_{\langle 2,*\rangle}) \subseteq c\{a,b\}^*$
This part follows from the construction for $L_\emptyset$ since $\tau_{\langle 1,*\rangle}(c) \vdash_{L_\emptyset} \tau_{\langle 2,*\rangle}(c)$ as follows :

$$\dfrac{S \,/\, A \vdash_{L_\emptyset} (S \,/\, A) \quad \vdots \vdash_{L_\emptyset} A}{S \,/\, A \vdash_{L_\emptyset} (S \,/\, A) \bullet A}$$

hence $\tau_{\langle 1,*\rangle}(w) \vdash_{L_\emptyset} \tau_{\langle 2,*\rangle}(w)$ and $\tau_{\langle 2,*\rangle}(w) \vdash_L S$ implies $\tau_{\langle 1,*\rangle}(w) \vdash_{L_\emptyset} S$ ■

5 Conclusion and remarks

5.1 Non-learnability for subclasses.

From the constructions in previous sections we get the following proposition as a corollary :

Proposition 3 (non-learnability) *The class of languages of rigid (or k-valued for an arbitrary k) Lambek grammars admits a limit point ; the class of rigid (or k-valued for an arbitrary k) Lambek grammars is not learnable from strings.*

Note. Our result has shown that in contrast to classical categorial grammars, rigid and k-valued Lambek grammars are not learnable from strings. This result holds for variants of Lambek calculus : (i) rigid Lambek grammars for $L_\emptyset$ (allowing empty sequence) using only $/$ (unidirectional) ; (ii) rigid Lambek grammars for L (with the non-empty left requirement) using $(/, \bullet)$. Our proof consists in the construction of limit points in each class : for (i) and (ii) we have $c\{a,b\}^*$ as limit point that are indeed rigid lambek grammars as already shown. In the first construction, we may also consider grammars of a bounded order.

5.2 Further remarks.

A stricly infinite chain of types. In each construction, the types $D_n = D_{\langle i,n\rangle}$ for c are such that : $... \vdash D_n \vdash D_{n-1} \vdash ... \vdash D_0$. This is the key point for the inclusion of languages. Moreover $D_{n-1} \not\vdash D_n$, which is reflected by the strictness of inclusion of languages. Our construction has thus exhibited a stricly infinite chain of types with respect to deduction for each version of Lambek calculus.

Future work. We are also interested in other variants of categorial grammars, such as the non-associative version of Lambek calculus or Multi-modal categorial grammars for which the construction of *rigid limit points* could be extended.

We also plan to consider not only the string languages but some structure languages in a grammatical inference perspective. In (Bonato and Retoré, 2001), it has been shown that rigid Lambek grammars are learnable in the limit from stuctured sentences. Such structures, that are studied in (Tiede, 1999), are complete

proof tree structures in normal forms. In our result, we show that, without any structure, some classes of Lambek grammars are not learnable in the limit from positive examples. It will now be very interesting to find an intermediate structure, that is more linguistically realistic than complete proof tree structures but that preserves the learnability in the limit of Lambek grammars.

References

Y. Bar-Hillel. 1953. A quasi arithmetical notation for syntactic description. *Language*, 29:47–58.

Roberto Bonato and Christian Retoré. 2001. Learning rigid lambek grammars and minimalist grammars from structured sentences. *Third workshop on Learning Language in Logic, Strasbourg.*

W. Buszkowski. 1997. Mathematical linguistics and proof theory. In van Benthem and ter Meulen (van Benthem and ter Meulen, 1997), chapter 12, pages 683–736.

Claudia Casadio. 1988. Semantic categories and the development of categorial grammars. In R. Oehrle, E. Bach, and D. Wheeler, editors, *Categorial Grammars and Natural Language Structures*, pages 95–124. Reidel, Dordrecht.

Jean-Yves Girard. 1995. Linear logic: its syntax and semantics. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Notes*, pages 1–42. Cambridge University Press.

E.M. Gold. 1967. Language identification in the limit. *Information and control*, 10:447–474.

Makoto Kanazawa. 1998. *Learnable classes of categorial grammars.* Studies in Logic, Language and Information. FoLLI & CSLI. distributed by Cambridge University Press.

Joachim Lambek. 1958. The mathematics of sentence structure. *American mathematical monthly*, 65:154–169.

Michael Moortgat. 1997. Categorial type logic. In van Benthem and ter Meulen (van Benthem and ter Meulen, 1997), chapter 2, pages 93–177.

Jacques Nicolas. 1999. Grammatical inference as unification. Rapport de Recherche RR-3632, INRIA. http://www.inria.fr/RRRT/publications-eng.html.

Christian Retoré. 2000. Systèmes déductifs et traitement des langues:un panorama des grammaires catégorielles. Technical Report RR-3917 2000, INRIA, Rennes, France. A revised version to appear in Traitement automatique du langage naturel, TSI.

Hans-Jörg Tiede. 1999. *Deductive Systems and Grammars: Proofs as Grammatical Structures.* Ph.D. thesis, Illinois Wesleyan University.

J. van Benthem and A. ter Meulen, editors. 1997. *Handbook of Logic and Language.* North-Holland Elsevier, Amsterdam.