Categorial Module Grammars of Bounded Size have Finite Bounded Density

Camilo Thorne

KRDB Centre
Free University of Bozen-Bolzano
3, Piazza Domenicani
39100 - Bolzano, Italy
cthorne@inf.unibz.it
http://www.inf.unibz.it/~thorne

Abstract. Categorial module grammars are a class of categorial grammars where in place of the lambek calculus (**L**) as type system, and types, we associate modules to words, that is, linear logic (**LL**) partial proof structures. However, their learnability conditions w.r.t. Gold's model remain, as yet, unknown. We provide a definition of this interesting class and show that relevant subclasses of it, most notably those of the grammars whose size is bounded by some $n \in \mathbb{N}$ and minimal for a given sample, are learnable in Gold's model (by identification in the limit) by applying Shinohara's theorem on the finite bounded density of their grammar system.

Keywords: Categorial grammar, learning in Gold's model, formal syntax, linear logic, proof nets.

1 Motivation

Categorial grammars ($\mathcal{CG}$) are a class of grammars that provide a mathematically elegant model of natural language syntax and semantics (cf. [3]), showing many advantages w.r.t. phrase structure grammars, like, for example, a clear link among syntax and semantic composition. Their links to linguistic theory have been thoroughly studied ever since R. Montague propounded in the 70's type theory or higher order logic as a means for modelling English semantics compositionally[1].

More recently, people have tackled the problem of their learnability, both symbolic and statistical. In the case of the former, of symbolic learnability, a well-known framework that has been well studied is Gold's model (cf. [4, 8]). It is thus known that categorial grammars cannot be learned from bare strings, but from structure languages, i.e. from samples where we consider not only the strings (the sentences or utterances of a speaker), but their parse trees. Kanazawa in [8] looks at all these problems in detail. Béchet and Foret in [2] extend these results to some classes of lambek grammars, based on the lambek calculus.

[1] I would like to thank Mr. D. Béchet for his advice and comments.

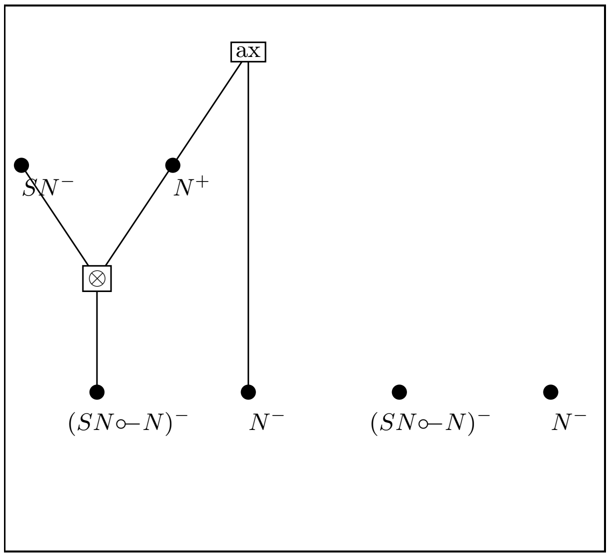

Fig. 1. *Three modules with tags. The first one of tag $(SN\multimap N)^-, N^-$ and the other two of tags $(SN\multimap N)^-$ and N^-, respectively.*

For, indeed, one can define alternatively categorial grammars by means of the lambek calculus (**L**) (cf. Lambek in [9]), a sequent calculus that was later shown to belong of the linear logic (**LL**) family intially defined by Girard in [6, 7]. This result has been thorougly studied in the field of mathematical linguistics by people like Retoré in [11, 10, 5]. When moving to the paradigm of lambek grammars, parse trees become undirected graphs: *proof nets*. One of the possible ways of constructing these proof nets is by building them incrementally from a finite set of elementary graphs called *modules* (cf. [1, 12]). Parsing consists in joining these graphs together through glue graphs called *axiom links* until a proof net is obtained. Hence, a good way of studying the learnability in Gold's model of lambek grammars would be to see what happens when we consider strings enriched with syntactic information by way of partial proof nets, i.e. modules. The formulas attached to a module are called its *tags* or *output types* (see Figure 1). But then, learnability results become quite blurred since the proofs in [8] hold mainly for categorial grammar of a more restricted expressive power, and not *prima facie* for those based on linear logic. This is a shortcoming worth remedying, due to the nice properties of sequent calculi and in particular of **LL**.

It is our aim in this paper to contribute in tackling this problem by proposing a class of categorial grammars, module grammars, where, in place of types, lexical items are assigned modules encoding their syntactic category. We will in particular show that relevant subclasses of module grammars, most notably those of the grammars whose size is bounded by some $n \in \mathbb{N}$ and that are reduced

$$\begin{aligned}\text{bounded size} + \text{containment} &\Rightarrow \text{bounded finite density}\\ &\Rightarrow \text{finite elasticity}\\ &\Rightarrow \text{learnability}\end{aligned}$$

Fig. 2. *Entailment of properties.*

w.r.t. a positive sample, are learnable in Gold's model (by identification in the limit) by applying Shinohara's theorem regarding the finite bounded density of their grammar system (cf. [14, 13]). This comes from the fact that a notion of *size* can be defined over module grammars, together with a *containment* relation or ordering.

The outline of this paper is as follows. We will begin by recalling some of the basic notions and properties of Gold's model essential for the understanding of the results. Next, we will proceed to define the class of categorial module grammars, illustrating it with some examples. The following section will be devoted to the learnability results. Finally, we will conclude with some general observations and by pinpointing some remaining open problems.

2 Learning in Gold's Model

In this section we recall the basics of Gold's model, that is, learning by identification to the limit from positive examples (cf. [8]). In so doing we will follow Shinohara's steps in [14, 13], introducing the notions essential to the proofs. This is schematized in Figure 2. The idea in this context is to: (i) Define a *grammar system* over an alphabet Σ. (ii) Define a *learning algorithm* (usually partial) for the language class generated by the grammars of this system (cf. [8]). Furthermore, grammars systems, as well as grammar and language classes can exhibit some structural properties that entail learnability in Gold's model and the existence of this algorithm: (i) *finite elasticity* and (ii) *bounded finite density.* Language classes are assumed to contain, exclusively, decidable languages.

Definition 1. (Grammar System) *A* grammar system *over an alphabet Σ (a finite set of symbols) is a triple $\mathcal{GS} = \langle\, \Sigma^*, \mathcal{G}, L \,\rangle$ where:*

1. *Σ^* is the set of finite strings over Σ.*
2. *$\mathcal{G}$ a grammar class over Σ.*
3. *$L \colon \mathcal{G} \to \mathcal{P}(\Sigma^*)$ is the* naming *function (associating a language, that is, a subset of Σ^*, to each grammar).*

A grammar class $\mathcal{G}$ from a grammar system $\mathcal{GS}$ is then learnable if there exists a (partial) algorithm $\phi_{\mathcal{G}}$ s.t. whenever we give it as input a sample $\mathcal{E} \subseteq L$ of an arbitary $L \in \mathcal{L}(\mathcal{G})$ (the class of languages generated by the grammars in $\mathcal{G}$), it will output its generating grammar G_L. Moreover, it should proceed as follows: it should scan, iteratively (by means of an unbounded loop) ever increasing finite sequences of samples, outputing succesive grammars, until a certain size $n_0 \in \mathbb{N}$ is reached, and converge thereafter, if the language is learnable, or run forever (cf. [8] for details). It is hence crucial to determine conditions that ensure its being total and that it always terminates, i.e. that this integer exists. This is what ensures the property of *finite elasticity* holding over the generated language class. That the algorithm exists and that it learns the grammars in the grammar class by identification in the limit. Formally:

Definition 2. (Learnability) *Let $\mathcal{GS} = \langle\, \Sigma^*, \mathcal{G}, L\,\rangle$ be a grammar system. The class $\mathcal{G}$ is said to be* learnable in Gold's model *or* learnable by identification in the limit from positive examples *iff a learning algorithm $\phi_{\mathcal{G}}$ exists for $\mathcal{G}$.*

Definition 3. (Infinite Elasticity) *A class $\mathcal{L}$ of langages over an alphabet Σ has* infinite elasticity *(*finite *otherwise) iff there exist two infinite sequences $\langle s_i \rangle_{i \in \mathbb{N}}$ and $\langle L_i \rangle_{i \in \mathbb{N}}$ with for all $i \in \mathbb{N}$, $L_i \in \mathcal{L}$, $s_i \in \Sigma^*$ and:*

1. *$s_i \notin L_i$.*
2. *$\{s_0, ..., s_i\} \subseteq L_{i+1}$.*

Theorem 1. (Angluin) *Let $\mathcal{GS} = \langle\, \Sigma^*, \mathcal{G}, L\,\rangle$ be a grammar system for which the word problem is decidable. Then, if $\mathcal{L}(\mathcal{G})$ has a finite elasticity, $\mathcal{G}$ is learnable.*

The idea of Shinohara in [14, 13] was that of exploiting the fact that given two things: (i) a finitary notion of *size* and (ii) a *containment* relation over a class of grammars, certain subclasses of it become learnable. Containment gives way to *reduced grammars*, that is, the minimal grammars generating the sample. The property of *finite bounded density* says then essentially that classes of reduced grammars of a bounded finite size are finite and hence learnable, because they generate a finite class of languages for which finite elasticity holds. Reduced grammars contain, intuitively, no redundancies.

Definition 4. (Reduced Grammar) *Let $G \in \mathcal{G}$. G is* reduced w.r.t. $\mathcal{X} \subset \Sigma^*$ *iff*

1. *$\mathcal{X} \subset L(G)$.*
2. *For all $G' \in \mathcal{G}$, if $G' \sqsubset G$ then $\mathcal{X} \nsubseteq L(G')$. Where $\sqsubset$ denotes an ordering relation among grammars called* containment.

Definition 5. (Bounded Finite Density) *Let $\mathcal{GS} = \langle\, \Sigma^*, \mathcal{G}, L\,\rangle$ be a grammar system. We say that $\mathcal{GS}$ has* bounded finite density *iff*

1. *For all $G, G' \in \mathcal{G}$, if $G \sqsubseteq G'$ then $L(G) \subseteq L(G')$. (i.e. L is a monotone increasing function), where $\sqsubseteq$ denotes the loose order associated to containment.*
2. *For all $\mathcal{X} \subset \Sigma^*$, for all $n \in \mathbb{N}$, $card\{L(G) \mid t(G) \leq n,\ G \in \mathcal{G}$ and G is reduced w.r.t. $\mathcal{X}\}$ is finite. Where t is some function computing the* size *of a grammar G.*

Theorem 2. (Shinohara 1990) *Let $\mathcal{GS} = \langle\, \Sigma^*, \mathcal{G}, L\,\rangle$ be a grammar system. If $\mathcal{GS}$ has bounded finite density, then the class $\{L(G) \mid G \in \mathcal{G}$ and $t(G) \leq n\}$ has finite elasticity and is learnable.*

In other words, classes of grammars of a bounded finite size yield (or engender) classes of languages with finite elasticity, granting the existence of a learning algorithm for them.

3 Categorial Module Grammars

In this section we formally define the class of categorial module grammars, and divide it in two: the grammars of finite alphabet and the grammars of infinite alphabet. Note that even in the latter case they remain finitary objects, since their typing function (which completely determines them) is defined always over a finite domain. This allows the definition of the two key notions: (i) The notion of *size*, which will be defined as the cardinality of the typing function's domain. This feature is crucial in proving the bounded finite density of their associated grammar system – and hence one the main results of the following section. (ii) The notion of *containment* among grammars, defined in terms of the set inclusion relationship among the domains of typing functions (called *supports*).

3.1 The Grammars

Definition 6. (Module Grammars) *A* categorial module grammar *or* module grammar *is a tuple $G = \langle\, \Sigma_{mod}, \delta_{mod}, supp(G), S\,\rangle$ where:*

1. *$\Sigma_{mod} \subseteq \Sigma \times \mathcal{M}$, called* module alphabet*, where Σ is a finite alphabet and $\mathcal{M}$ a possibly infinite set of modules.*
2. *$\delta_{mod} \colon \Sigma_{mod} \to \mathcal{P}_f(\mathcal{M})$ is a partial* typing *function onto the finite subsets of $\mathcal{M}$ s.t. $\delta_{mod}(\langle u, M\rangle) = \{M\}$ whenever it is defined on $\langle u, M\rangle$.*
3. *$supp(G) \subset \Sigma_{mod}$ is the domain of δ_{mod}, a finite set called the* support *of G.*
4. *S is the type of recognized strings.*

As with other kinds of categorial grammars, whenever $M \in \delta_{mod}(\langle u, M\rangle)$ we write $G : \langle u, M\rangle \mapsto M$. The typing function of a module grammar basically projects the module that tags the tokens in Σ. Note that a token $u \in \Sigma$ can be tagged with an infinite number of pairwise distinct modules: Σ_{mod} can be an infinite set. Module grammars are meant to recognize or generate words enriched with the syntactic information encoded by modules. The intuition behind the fact that their typing function is partial is that they, so to speak, individuate the tokens and the syntactic categories (i.e., the modules) relevant for the ensuing derivations and parses of sentences of the language they are meant to recognize.

The language engendered or recognized by a module grammar is the set of sentences for which, when we glue together the modules that "decorate" words, a proof net can be obtained. The proof net thus takes the place of a derivation or proof tree, as we said in the introduction.

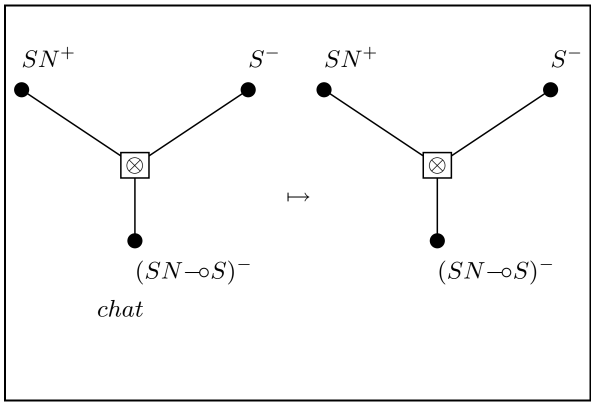

Fig. 3. *A typing rule of G_s.*

Definition 7. (Recognized Language) *The* language recognized *by a module grammar G, denoted $L_r(G)$, is the set of strings $s = u_1...u_n \in \Sigma^*$ s.t. $\langle u_0, M_0\rangle...\langle u_n, M_n\rangle \in (\Sigma_{mod})^*$, δ_{mod} is defined on $\langle u_i, M_i\rangle$, for $i \in [1,n]$, and there is a proof net that can be built out of the modules $M_0, ..., M_n$.*

Moreover, for every sequence (or string) $s = \langle u_0, M_0\rangle...\langle u_n, M_n\rangle \in (\Sigma_{mod})^*$ we call each $\langle u_i, M_i\rangle$, for $i \in [1,n]$ a *factor* of s.

Definition 8. (Grammar Containment) *Let $G, G' \in \mathcal{G}_{mod}$ (resp. $\mathcal{G}_{modf}$). $G \sqsubseteq G'$ iff $supp(G) \subseteq supp(G')$. We denote $\sqsubset$ the associated strict order.*

Definition 9. (Size of a Module Grammar) *Let $G \in \mathcal{G}_{mod}$ (or $\mathcal{G}_{modf}$). The size of G, that is $t(G)$, is equal to the cardinality of $supp(G)$, i.e. $t(G) = card(supp(G))$.*

Definition 10. (Subclasses) *For simplicity, we divide the subclasses of the class of module grammars as follows:*

1. *A* module grammar of finite alphabet *is a module grammar over a module alphabet such that Σ_{mod} is finite. Otherwise it is a* module grammar of infinite alphabet*.*
2. *A* module grammar of bounded size *over a module alphabet $\Sigma \times \mathcal{M}$ is a module grammar of size $\leq n$, for $n \in \mathbb{N}$ and reduced w.r.t. a finite sample $\mathcal{X} \subset \Sigma \times \mathcal{M}$.*

We denote respectively, $\mathcal{G}_{modf}$, $\mathcal{G}_{mod}$ and $\mathcal{G}^n_{mod}$, for $n \in \mathbb{N}$, these classes. $\mathcal{L}(\mathcal{G}_{modf})$, $\mathcal{L}(\mathcal{G}_{mod})$ and $\mathcal{L}(\mathcal{G}^n_{mod})$, for $n \in \mathbb{N}$, denoting, respectively, their associated language classes.

Definition 11. (Grammar Systems) *The grammar system for module grammars of infinite alphabet is:*

$$\mathcal{GS}_{mod} = \langle\, (\Sigma_{mod})^*,\, \mathcal{G}_{mod},\, L\,\rangle$$

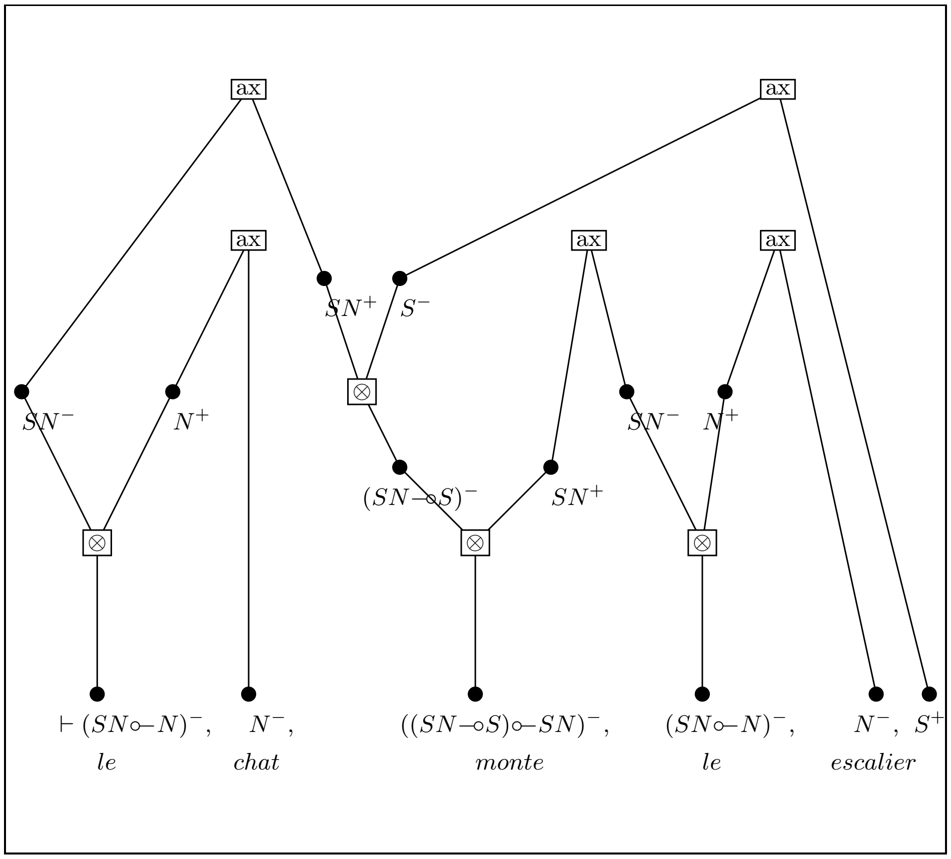

Fig. 4. *A grammatical sentence recognized by* G_s

Grammar systems are defined analogously for module grammars of finite alphabet and for module grammars of bounded size.

3.2 A Sample Module Grammar

In order to illustrate the notions introduced above, let us look at an example of a module grammar covering a fragment of French. Consider grammar $G_s = \langle\, \Sigma_{mod},\, \delta_{mod},\, sup(G_s),\, S\,\rangle$, where δ_{mod} is defined by rules of the form seen in Figure 3.

According to this grammar, the sentence *le chat monte l'escalier* $\in L(G_s)$ since there is a proof-net for the sequent:

$$\vdash (SN\circ\!\!- N)^-, N^-, ((SN\multimap S)\circ\!\!- SN)^-, (SN\circ\!\!- N)^-, N^-, S^+$$

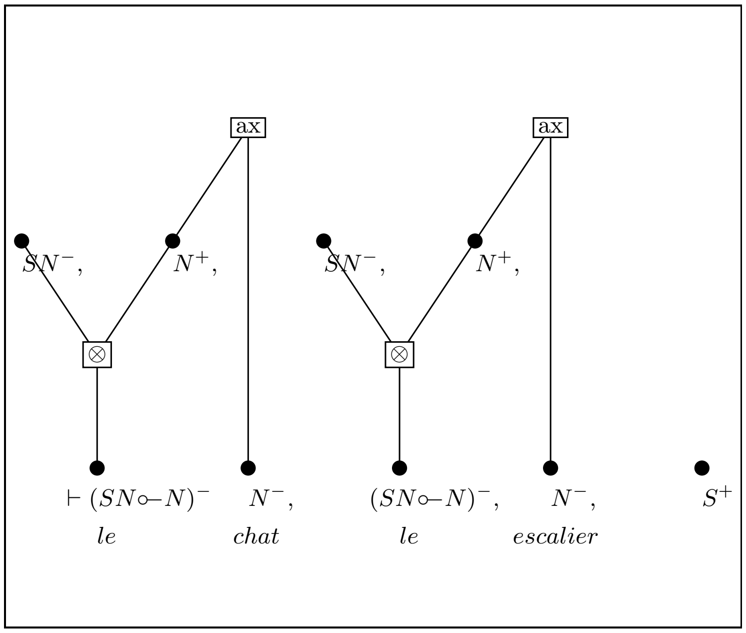

Fig. 5. *An ungrammatical sentence not recognized by G_s.*

This is because all the correction graphs of the proof structure or proof graph are acyclic and connected. This implies its being sequentialisable and, therefore, a proof net. See Figure 4.

On the other hand, the sentence *le chat l'escalier* $\notin L(G_s)$, because no proof net can be constructed for the sequent:

$$\vdash (SN \circ\!- N)^-, N^-, (SN \circ\!- N)^-, N^-, S^+$$

No other proof structure than the one seen in Figure 5 exists.

The size of the support of a module grammar depends on the size of Σ (on the words, regardless of the modules attached to them, that can be infinitely many). As Σ is always finite, so too is the support. The size of G_s is $card(supp(G_s)) = 5$ (the number of words seen in the example). Note that we assume a typing rule per word.

4 Some Positive and Negative Results

In this section we present some positive and negative results regarding the learnability of module grammars in Gold's model. The reasoning, for the main result, namely, learnability for module grammars of a size bounded by an $n \in \mathbb{N}$, will closely follow the outline provided by Figure 2. That is, we will prove, thanks to the properties of containment and size that (i) their grammar system has a finite bounded density, entailing learnability by identification on the limit (cf. **Theorem 4**) of these subclasses, that (ii) infinite elasticity holds for the whole class (**Theorem 5**) and that (iii) module grammars of finite alphabet are learnable in Gold's model (**Theorem 3**).

Lemma 1. *The class $\mathcal{L}(\mathcal{G}_{modf})$ has finite elasticity.*

Proof. Suppose the contrary. There exist then two infinite sequences $\langle L_i\rangle_{i\in\mathbb{N}}$ and $\langle s_i\rangle_{i\in\mathbb{N}}$ such that for all $i\in\mathbb{N}$, $L_i\in\mathcal{L}(\mathcal{G}_{modf})$, $s_i\in(\Sigma_{mod})^*$ and:

(**1**) $s_i\notin L_i$;
(**2**) $\{s_0,...,s_i\}\subseteq L_{i+1}$.

Now, Σ_{mod} is finite. Thus the class $\mathcal{G}_{modf}$ is finite too and *a fortiori*, $\mathcal{L}(\mathcal{G}_{modf})$ as well. Let $\Sigma_{mod}=\{\langle u_1,M_1\rangle,...,\langle u_k,M_k\rangle\}$. Module grammars are determined by their typing function. Thus, for all $i\in\{1,...,k\}$, for all $G\in\mathcal{G}_{modf}$:

$$\delta_{modf}(\langle u_i,M_i\rangle)=\{M_i\}.$$

That is, the grammars of the class differ according to their typing function's being defined on the factor $\langle u_i,M_i\rangle$, for $i\in\{1,...,k\}$. This gives us as many grammars as subsets of Σ_{mod}, i.e., 2^k, a finite number. And if the grammar class is finite, the same is true for the corresponding language class. This implies that there exist $i,j\in\mathbb{N}$, $i\neq j$, such that $L_i=L_j$. Hence, assume, w.l.g. that $i<j$. Then $i+1\leq j$. Then, by (**1**), $s_i\in L_j=L_i$. But, by (**2**), $s_i\notin L_i$. Contradiction. **Q.E.D.**

Theorem 3. *The class $\mathcal{G}_{modf}$ is learnable.*

Proof. Corrollary of **Lemma 1** by applying **Theorem 1**. **Q.E.D.**

Lemma 2. *For every $n\in\mathbb{N}$, the system $\mathcal{GS}^n_{mod}$ has bounded finite density.*

Proof. We verify each condition at a time:

– For every $n\in\mathbb{N}$, for every $G,G'\in\mathcal{G}^n_{mod}$, $G\sqsubseteq G'$ implies $L(G)\subseteq L(G')$ (trivial).
– Let $\mathcal{X}\subseteq(\Sigma_{mod})^*$, $\mathcal{X}$ be finite. Put:

$$\mathcal{R}_n=\{L(G)\mid G\in\mathcal{G}_{mod},\ t(G)\leq n\text{ and }G\text{ is reduced w.r.t. }\mathcal{X}\}.$$

Then for any $n\in\mathbb{N}$, $\mathcal{R}_n$ is a finite class. We prove this by induction on $n\in\mathbb{N}$:
- ($n=0$). $\mathcal{R}_0$ contains only one grammar, namely, the empty module grammar, i.e. the module grammar G such that $card(supp(G))=0$.
- ($n=k+1$). By induction hypothesis the property is true up to k. Thus $\mathcal{R}_k$ is a finite class. Note that $\mathcal{R}_{k+1}=\mathcal{L}(\mathcal{G}^{k+1}_{mod})$. We claim that $\mathcal{R}_{k+1}-\mathcal{R}_k$, i.e. $\mathcal{L}(\mathcal{G}^{k+1}_{mod}-\mathcal{G}^k_{mod})$ is finite. This is because $\mathcal{G}^{k+1}_{mod}-\mathcal{G}^k_{mod}$ is either empty, a singleton class or a class of equivalent grammars, i.e. of grammars

generating exactly the same language. Otherwise, there are at least two module grammars $G, G' \in \mathcal{G}^{k+1}_{mod} - \mathcal{G}^{k}_{mod}$ such that $L(G) \neq L(G')$ which implies that $supp(G) \neq supp(G')$. This means that for some $\langle u, M\rangle$, $\langle u, M\rangle \in supp(G')$ and $\langle u, M\rangle \notin supp(G)$. Now, G' is reduced w.r.t. $\mathcal{X}$ by definition of $\mathcal{R}_{k+1}$. Thus $\mathcal{X} \subset L(G')$ and hence $\langle u, M\rangle$ is a factor of some (structure) string s in $\mathcal{X}$. But, since G is also reduced also w.r.t. $\mathcal{X}$, then $\mathcal{X} \subset L(G)$ holds as well, whence it follows that G's typing function is defined on all of the factors of the strings of $\mathcal{X}$, and thus on $\langle u, M\rangle$, i.e. $\langle u, M\rangle \in supp(G)$. Contradiction. Thus $\mathcal{R}_{k+1} = \mathcal{R}_k \cup (\mathcal{R}_{k+1} - \mathcal{R}_k)$ is finite too, fact that closes the proof. **Q.E.D.**

Theorem 4. *For every $n \in \mathbb{N}$, the class $\{L(G) \mid G \in \mathcal{G}_{mod}$ and $t(G) \leq n\}$ has finite elasticity and is learnable.*

Proof. It is an immediate consequence of **Lemma 2** by application of **Theorem 2**. **Q.E.D.**

Theorem 5. *The class $\mathcal{L}(\mathcal{G}_{mod})$ has infinite elasticity.*

For the proof that follows we adopt the convention of representing a module M by its tag A. Thus, a module M having as tag $SN{-\!\circ}(S{\circ\!-}SN)$ will be written $SN{-\!\circ}(S{\circ\!-}SN)$.

Proof. Let us proceed by steps: Let us define first a sequence $\langle S^i\rangle_{i\in\mathbb{N}}$ of modules:

$$S^0 = S^-;$$

$$S^1 = (S{-\!\circ}S)^-;$$

$$S^i = (\underbrace{S{-\!\circ}(...(S}_{i-times}{-\!\circ}S)))^-.$$

And then two infinite sequences $\langle s_i\rangle_{i\in\mathbb{N}}$, $\langle L_i\rangle_{i\in\mathbb{N}}$ such that, for $i \in \mathbb{N}$, $L_i \in \mathcal{L}(\mathcal{G}_{mod})$ and $s_i \in (\Sigma_{mod})^*$. $\langle s_i\rangle_{i\in\mathbb{N}}$ is the sequence defined by:

$$s_0 = \langle u, S^0\rangle;$$

$$s_1 = \langle u, S^0\rangle\langle u, S^1\rangle;$$

$$s_i = \underbrace{\langle u, S^0\rangle...\langle u, S^0\rangle}_{i-times}\langle u, S^i\rangle.$$

Where the module S^i is the i-est term of the sequence $\langle S^i\rangle_{i\in\mathbb{N}}$. We can remark, also, that for any $i, j \in \mathbb{N}$, if $i \neq j$, then $\langle u, S^i\rangle \neq \langle u, S^j\rangle$ (and *a fortiori* $S_i \neq S_j$). While $\langle L_i\rangle_{i\in\mathbb{N}}$ is defined by means of the sequence $\langle G_i\rangle_{i\in\mathbb{N}}$ of their generating grammars, namely the sequence where: G_0 is the grammar such that:

$$supp(G_0) = \emptyset.$$

G_1 is the grammar such that:

$$G_1 : \langle u, S^0 \rangle \mapsto S^0;$$

G_i is the grammar such that:

$$G_i : \langle u, S^0 \rangle \mapsto S^0;$$

$$G_i : \langle u, S^1 \rangle \mapsto S^1;$$

$$\vdots$$

$$G_i : \langle u, S^i \rangle \mapsto S^i.$$

We then put: $L_i = L(G_i)$, for $i \in \mathbb{N}$. We can now claim that for every $n \in \mathbb{N}$:

(1) $s_n = \overbrace{\langle u, S^0 \rangle ... \langle u, S^0 \rangle}^{n-times} \langle u, S^n \rangle \notin L_n = L(G_n)$ and
(2) $\{s_0, ..., s_n\} \subseteq L_{n+1} = L(G_{n+1})$.

Claims **(1)** and **(2)** are proven simultaneously by induction on $n \in \mathbb{N}$:

– $(n = 0)$ The basis is trivial: $s_0 = \langle u, S^- \rangle \notin L(G_0)$ (since $\langle u, S^- \rangle \notin supp(G_0)$), but $\{s_0\} \subseteq L(G_1)$.

– $(n = k+1)$. The property is true for k. Now, $s_{k+1} = \overbrace{\langle u, S^0 \rangle ... \langle u, S^0 \rangle}^{(k+1)-times} \langle u, S^{k+1} \rangle$. Furthermore, we have that, on the one hand, $\langle u, S^{k+1} \rangle \notin supp(G_{k+1})$. Therefore $s_{k+1} \notin L(G_{k+1})$, since the sequent

$$\vdash \overbrace{S^-, ..., S^-}^{(k+1)-times}, S^+$$

Admits no proof. And on the other hand $\langle u, S^{k+1} \rangle \in supp(G_{k+2})$ and the sequent

$$\vdash \overbrace{S^-, ..., S^-}^{(k+1)-times}, (\underbrace{S \multimap (...(S}_{(k+1)-times} \multimap S)))^-, S^+$$

Admits a proof. Hence $s_{k+1} \in L(G_{k+2})$. Now, by induction hypothesis on k, $\{s_0, ..., s_k\} \subseteq L(G_{k+1}) \subseteq L(G_{k+2})$ and thus $\{s_0, ..., s_k, s_{k+1}\} \subseteq L(G_{k+2})$ too. **Q.E.D.**

5 Concluding Remarks

Summing up, in this paper we have shown:

– That the class $\mathcal{L}(\mathcal{G}_{modf})$, i.e. of languages generated by categorial module grammars of finite alphabet, has finite elasticity and hence $\mathcal{G}_{modf}$ is learnable in Gold's model.

– That, for all $n \in \mathbb{N}$, the grammar system $\mathcal{G}S^n_{mod} = \langle (\Sigma \times \mathcal{M})^*, \mathcal{G}^n_{mod}, L \rangle$, i.e. that of categorial module grammars of infinite alphabet, has bounded finite thickness and that hence, the class $\mathcal{G}^n_{mod}$ is learnable in Gold's model.
– That the class $\mathcal{L}(\mathcal{G}_{mod})$ has infinite elasticity

These results are quite general, and can be applied to module grammars where the modules encode also semantic information (since syntactic types are homeomorphic to semantic types). Moreover, it is easy to see that the limit point construction for proving unlearnability (see Kanazawa, cf [8]) doesn't hold for these grammars. For indeed, finding a language identical to the arbitrary union of an infinite chain of languages of $\mathcal{L}(\mathcal{G}_{mod})$ strictly contained one in each other would only be possible by defining a grammar with an infinite support, something that is precluded by definition. In any case, since we remain within the boundaries of **LL**, module grammars inherit all the nice properties that any lambek grammar should exhibit: close association among syntax and semantics, semantic compositionality, proof normalization, etc. However to prove that the whole class is actually learnable in Gold's model, a learning algorithm verifying the requirements stated in [8] should be defined. The negative result mentioned above (*viz.* infinite elasticity of module grammars) precludes our being able to prove the existence of a learning algorithm through purely structural properties.

References

1. Denis Bechet. Incremental parsing of lambek calculus using proof-net interfaces. http://www-lipn.univ-paris13.fr/∼bechet, 2002.
2. Denis Bechet et Annie Foret. k-valued non-associative lambek grammars are learnable from function-argument structures. *Electronic Notes in Theoretical Computer Science*, (84), 2003.
3. Bob Carpenter. *Type-Logical Semantics.* The MIT Press, 1997.
4. Antoine Cornuejols et Yves Kodratoff. *Apprentissage artificiel. Concepts et algorithmes.* Eyrolles, 2002.
5. Christian Retore et François Lemarche. Proof nets for the lambek calculus – an overview. In *Proceedings of the 1996 Roma Workshop 'Proofs and Linguistic Categories – Applications of Logic to the Analysis and Implemenation of Natural Language'*, 1996.
6. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102.
7. Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types.* Number 7 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1988.
8. Makoto Kanazawa. *Learnable Classes of Categorial Grammars.* CSLI, 1998.
9. Joachim Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170, 1958.
10. Christian Retore. Calcul de lambek et logique linéaire. *T.A.L.*, 37(2):39–70.
11. Christian Retore. Logique linéaire et syntaxe des langues. Technical report, Université de Nantes et INRIA, 2002.
12. Christian Retore et André Lecomte. Words as modules and modules as partial proof-nets in a lexicalised grammar. In Carlos Martin-Vide, editor, *Mathematical and Computational Analysis of Natural Language*, volume 45 of *Functional and Structural Linguistics.* John Benjamins, 1998.

13. Takeshi SHINOHARA. Inductive inference from positive data is powerful. In *Annual Workshop on Computational Learning Theory*, volume 3, 1990.
14. Takeshi SHINOHARA. Inductive inference of monotonic formal systems from positive data. In *International Workshop on Algorithmic Learning Theory*, number 1, 1990.