MEANING HELPS LEARNING SYNTAX*

Isabelle Tellier

LIFL and Université Charles de Gaulle-lille3 (UFR IDIST)
59 653 Villeneuve d'Ascq Cedex, FRANCE
Email : tellier@univ-lille3.fr

Abstract. In this paper, we propose a new framework for the computational learning of formal grammars with positive data. In this model, both syntactic and semantic information are taken into account, which seems cognitively relevant for the modeling of natural language learning. The syntactic formalism used is the one of Lambek categorial grammars and meaning is represented with logical formulas. The principle of compositionality is admitted and defined as an isomorphism applying to trees and allowing to automatically translate sentences into their semantic representation(s). Simple simulations of a learning algorithm are extensively developed and discussed.

1 Introduction

Natural language learning seems, from a formal point of view, an enigma.

As a matter of fact, every human being, given nearly exclusively positive examples ([25]), is able at the age of about five to master his/her mother tongue. Though every natural language has at least the power of context-free grammars ([22]), this class is not computationally learnable with positive data in usual models ([9, 24]).

How can a formal theory of learning give account of such a success ? Various solutions have been proposed. Following the Chomskian intuitions ([4, 5]), it can be admitted that natural languages belong to a restricted family and that the human mind includes an *innate knowledge* of the structure of this class. For example, context-sensitive grammars become learnable with positive data if the learner knows a bound on the number of rules in the grammar ([24]).

Another approach consists in putting structural, statistical or complexity constraints on the *examples proposed to the learner*, making his/her induction easier ([15, 21]). This solution formalizes the help provided by a professor ([6]). A particular family of research, more concerned with the cognitive relevance of its models, considers that learning a natural language is very different from learning a formal language, because in *natural* situations, examples are always provided with semantic and pragmatic information ([10, 2, 14, 11]). This approach may be seen as

*This research was partially supported by "Motricité et cognition" : contrat par objectifs de la région Nord/Pas de Calais and basic ideas of this paper were presented at the Workshop on Paradigms and Grounding in Language Learning of the conference Computational Natural Language Leaning 98.

another interpretation of the previous ones. As a matter of fact, it implies that natural languages belong to the restricted family of languages with which *semantics compatible with our world* can be associated (even if this class is hardly characterizable), and it also assumes that examples provided to the learner are those which are semantically coherent. This is the family our research belongs to.

A fundamental property of natural languages is then taken into account here : their *meaningfulness*. But this property is computationally tractable only if we have at our disposal a theory that precisely articulates syntax and semantics. The strongest possible articulation is known as the Fredge's *principle of compositionality*, which states that the meaning of a sentence only depends on the meaning of its constituents and of their mode of combination. This principle has acquired an explicit formalization with the works of Richard Montague ([16, 7]) and his inheritors.

In this paper, we will first expose an up-to-date version of this syntactico-semantic framework, based on a type of grammars called *categorial grammars*, and we will then show how it can been used in a formal theory of natural language learning.

2 Syntactic analysis with categorial grammars

In categorial grammars, each member of the vocabulary is associated with a finite set of categories characterizing its combinatorial potentialities.

The syntax only depends on the information associated with each word. This strong lexicalization is well adapted to natural languages ([18]).

2.1 General definition of categorial grammars

A categorial grammar G is a 4-tuple G=<V, C, f, S> where :

- V is the finite alphabet (or vocabulary) of G;
- C is the finite set of basic categories of G;

We define the set of all possible categories of G, noted C' , as the closure of C for the fractional operators noted / and \. C' is the smallest set of categories verifying :

* $C \subseteq C'$;
* if $X \in C'$ and $Y \in C'$ then $X/Y \in C'$ and $Y\backslash X \in C'$;

- f is a function : V—>P(C') where P(C') is the set of finite subsets of C' , which associates each element v in V with the finite set $f(v) \subseteq C'$ of its categories;
- $S \in C$ is the axiomatic category of G.

The operators / and \ can be considered as oriented fractions. In this framework, the set of syntactically correct sentences is the set of finite concatenations of elements of the vocabulary for which there exists an assignment of categories that can be *reduced* to the axiomatic category S. There are different classes of categorial grammars according to the admitted *reduction rules*. We will use the most general one, called L-grammars.

2.2 L-grammars (from [13])

A Lambek grammar (or L-grammar) is a categorial grammar for which admitted reduction rules are all the valid sequents of the Lambek-Gentzen calculus defined by :

- an infinite number of axioms : for every category X in C', X —> X (A) is valid;
- two couples of inference rules among sequents, written in the Gentzen style (i.e. if the sequent(s) above the line is (are) valid, then the one under is also valid) :

$$*\ \frac{\Gamma \cdot X \longrightarrow Y}{\Gamma \longrightarrow Y/X}\ (R/) \qquad \frac{X \cdot \Gamma \longrightarrow Y}{\Gamma \longrightarrow X\backslash Y}\ (R\backslash)$$

$$*\ \frac{\Gamma \longrightarrow X \qquad \Delta \cdot Y \cdot \Phi \longrightarrow Z}{\Delta \cdot Y/X \cdot \Gamma \cdot \Phi \longrightarrow Z}\ (L/) \qquad \frac{\Gamma \longrightarrow X \qquad \Delta \cdot Y \cdot \Phi \longrightarrow Z}{\Delta \cdot \Gamma \cdot X\backslash Y \cdot \Phi \longrightarrow Z}\ (L\backslash)$$

where X, Y and Z are in $\in C'$ and Δ, Φ and Γ are concatenations of categories ($\Gamma \neq \varnothing$). The language L(G) defined by G is then :

$L(G)=\{w \in V^*;\ \exists n \in N\ \forall i \in \{1,..., n\}\ w_i \in V,\ w=w_1...w_n,\ \exists C_i \in f(w_i),\ C_1...C_n —*—> S\}$.

Useful valid sequents :

The most simple valid sequents, for any categories X and Y are the following :

R1 : X/Y . Y —> X R'1 : Y . Y\X —> X

These rules straightforwardly follow from (L/) and (L\) and explain, if the concatenation of categories is assimilated to a multiplication, the nature of / and \ (we use the notational variant of [17]). Other interesting valid sequents are ([17]) :

- R2 : X/Y . Y/Z —> X/Z R'2 : Z\Y . Y\X —> Z\X
- R3 : (Y\X)/Z —> Y\(X/Z) R'3 : Y\(X/Z) —> (Y\X)/Z
- R4 : X —> Y/(X\Y) R'4 : X —> (Y/X)\Y

Example grammar :

Let us define a L-grammar for the analysis of a small subset of natural language.

The vocabulary is V={a, every, man, woman, John, Paul, runs, loves, meets, is...}. The set of usual basic categories needed here is C={S, T, CN}. In this set, S is the axiomatic category, T stands for *term* and is assigned to proper names, while intransitive verbs receive the category T\S and transitive ones the category (T\S)/T. CN means *common noun* and determiners like *a* and *every* receive two categories : (S/(T\S))/CN and ((S/T)\S)/CN, depending on their position as a subject or as a direct object. This grammar allows to analyze simple sentences as seen in Fig. 1.

$$\frac{\dfrac{CN \longrightarrow CN\ (A) \qquad \dfrac{T\backslash S \longrightarrow T\backslash S\ (A) \qquad S \longrightarrow S\ (A)}{S/(T\backslash S) \cdot T\backslash S \longrightarrow S\ (L/)}}{S/(T\backslash S))/CN \cdot CN \cdot T\backslash S \longrightarrow S\ (L/)}$$

a man runs

$$\frac{\dfrac{T \longrightarrow T\ (A) \qquad \dfrac{T \longrightarrow T\ (A) \qquad T \cdot T\backslash S \longrightarrow S\ (L\backslash)}{S \longrightarrow S\ (A)}}{T \cdot (T\backslash S)/T \cdot T \longrightarrow S\ (L/)}$$

John is Paul

Fig. 1. analysis trees of simple sentences

Formal properties of L-grammars :

L-grammars have been deeply studied. The main results about them are :

- The Lambek-Gentzen calculus is decidable. As a matter of fact, in each of its inference rules, there is one more operator / or \ under the line than there are above. To decide if a given sequent is valid, one only has to test on it each possible inference

rule in backward chaining, eliminating its operators one after the other, until only axioms (if the sequent is valid) are left.

- It is impossible to define a finite set of valid sequents (for example, rules R1 to R'4) equivalent with the complete Lambek-Gentzen calculus ([26]).
- The class of languages defined by L-grammars is the class of context-free languages ([19]) and the membership problem can be solved in polynomial time ([8]).

3 From syntax to semantics

The key idea of Montague's work ([16]) was to define an isomorphism between syntactic trees and semantic ones.

This definition is the formal expression of the principle of compositionality. It allows to automatically translate sentences in natural language into logical formulas.

3.1 The semantic representation

The logic we will use, called IL, is a simplified version of the intensional logic defined by Montague ([16, 7]). It generalizes first order predicate logic by including typed lambda-calculus.

- IL is a typed language : the set I of all possible types of IL includes
  * elementary types : $e \in I$ (type of *entities*) and $t \in I$ (type of *truth values*);
  * for any types $u \in I$ and $v \in I$, $<u,v> \in I$ ($<u,v>$ is the type of functions taking an argument of type u and giving a result of type v).
- semantics : a denotation set D_w is associated with every type $w \in I$ as follows :
  * $D_e = E$ where E is the denumerable set of all entities of the world;
  * $D_t = \{0,1\}$;
  * $D_{<u,v>} = D_v^{D_u}$: the denotation set of a composed type is a function.

IL also includes usual quantifiers and lambda-expressions.

3.2 Translation as an isomorphism

Each analysis tree produced by a L-categorial grammar can be *translated* into IL :

- translation of the categories into logical types (function k : C' —> I) :
  * basic categories : k(S)=t and in our example : k(T)=e, k(CN)=<e,t>;
  * derived categories : for any $X \in C'$ and $Y \in C'$, $k(X/Y)=k(Y\backslash X)=<k(Y),k(X)>$.
- translation of the words (function q : $V \times C'$ —> IL) : each couple (v,U) where $v \in V$ and $U \in f(v) \subseteq C'$ is associated with a logical formula q(v,U) of IL of type $k(U) \in I$.

The most usual and useful translations are :

* $q(a,(S/(T\backslash S))/CN)=q(a,((S/T)\backslash S)/CN)=\lambda P \lambda Q \exists x[P(x) \wedge Q(x)]$
  $q(every,S/(T\backslash S))/CN)=q(every,((S/T)\backslash S)/CN)=\lambda P \lambda Q \forall x[P(x) \rightarrow Q(x)]$
  where x and y are variables of type e, P and Q variables of type <e,t>.
* the verb *to be*, as a transitive verb is translated by
  $q(be,(T\backslash S)/T)=\lambda x \lambda y[y=x]$ with x and y variables of type e.

* Every other word w is translated into a logical constant noted w'.

- translation of the rules of the Lambek-Gentzen calculus :
  * the axioms X —> X are translated by the identity function : λx.x;
  * the inference rules are respectively translated by rules applying to IL formulas :

$$\frac{t \,.\, x \text{ —> } y}{t \text{ —> } \lambda x[y]} \qquad \frac{x \,.\, t \text{ —> } y}{t \text{ —> } \lambda x[y]}$$

$$\frac{t \text{ —> } x \qquad u \,.\, f(x) \,.\, v \text{ —> } z}{u \,.\, f \,.\, t \,.\, v \text{ —> } z} \qquad \frac{t \text{ —> } x \qquad u \,.\, f(x) \,.\, v \text{ —> } z}{u \,.\, t \,.\, f \,.\, v \text{ —> } z}$$

As all rules Ri ($1 \le i \le 4$) can be proved in the Lambek calculus, the corresponding translation rules called Wi can be deduced from our translations ([17]) :

- W1 : f . a —> f(a) — W'1 : a . f —> f(a)
- W2 : f . g —> λa[f(g(a))] — W'2 : g . f —> λa[f(g(a))]
- W3 : f —> λaλb[(f(b))(a)] — W'3 : f —> λaλb[(f(b))(a)]
- W4 : a —> λf[f(a)] — W'4 : a —> λf[f(a)]

Examples

Fig. 2 displays the translation of the first tree of Fig. 1. The unknown translation of the sentence is first noted M. From the isomorphism, we obtain :

$M = \lambda P \lambda Q \exists x[P(x) \wedge Q(x)](man')(run') = \lambda Q \exists x[man'(x) \wedge Q(x)](run')$
$= \exists x[man'(x) \wedge run'(x)]$

$$\dfrac{\dfrac{T\backslash S \text{ —> } T\backslash S \;(A) \qquad S \text{ —> } S \;(A)}{CN \text{ —>} CN \;(A) \quad S/(T\backslash S).T\backslash S \text{ —>} S \;(L/)} \qquad \dfrac{run' \text{—>}_b \; \lambda P \lambda Q \exists x[P(x) \wedge Q(x)](a)(b) \text{—>} M}{man' \text{—>}_a \; P \lambda Q \exists x[P(x) \wedge Q(x)](a).run' \text{—>} M}}{(S/(T\backslash S))/CN \,.\, CN \,.\, T\backslash S \text{ —> } S \;(L/)} \quad => \quad \dfrac{}{\lambda P \lambda Q \exists x[P(x) \wedge Q(x)].man'.run' \text{—>} M}$$

a man runs

Fig. 2. translation of the first analysis tree of Fig. 1

Similarly, the second tree of Fig. 1 is translated into : [John' =Paul'. This system gives account of ambiguities : ambiguous sentences give rise to several different trees translated into several different formulas.

4 First approach to the learning model

Our purpose is to provide a computational model of natural language learning. Now that the linguistic framework is given, it remains to fix what is supposed to be known by the learner and what is to be learned, under which conditions.

4.1 Innate knowledge and concepts to be learned

When a human being learns a natural language, we suppose that he has at his disposal sentences syntactically correct and semantically relevant.

The corresponding situation in our model is an algorithm which takes as inputs a syntactically correct sentence together with its logical translation into IL.

The innate knowledge supposed is reduced to the inference rules of the Lambek calculus and the corresponding translation rules. As opposed to usual semantic-based methods of learning, *no word meaning is supposed to be initially known.*

Finally, what does the learner have to learn ? In our linguistic framework, all syntactic and semantic information are attached to the members of the vocabulary. More precisely, the knowledge to be learned can be represented as a finite list of triplets of the form (v,U,w) where $v \in V$, $U \in f(v) \subseteq C'$ and $w=q(v,U) \in IL$.

Example :

Learning the example grammar of 2.2 and its logical translation means learning the following set :

{ (a, (S/(T\S))/CN, λPλQ∃x[P(x)∧Q(x)]), (a, ((S/T)\S)/CN, λPλQ∃x[P(x)∧Q(x)]),
(man, CN, man'), (woman, CN, woman'), (John,T,John'), (Paul,T,Paul'),
(runs,T\S,run'),(meets,(S\T)/T,meet'),(is,(S\T)/T,λxλy[y=x])...}

Fig. 3. shows the components of our model.

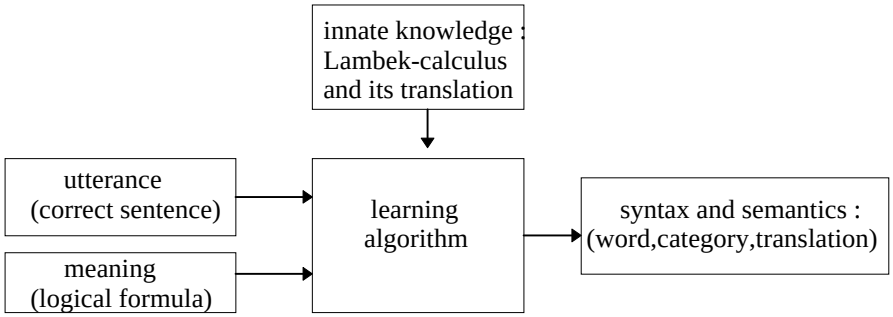


Fig. 3. the learning model

4.2 The learning algorithm

The learning strategy we propose is the following one :

- current hypothesis := ∅;
- For every couple <p, l> where p is a sentence and l is a logical formula do :
  - For every word in p do :
  * if it belongs to the current hypothesis, affect it the corresponding category;
  * else affect it the categories allowing the analysis of p;
  - For every possible analysis tree for p do :
    - for every couple (word category) in this tree do :
    * if it belongs to the current hypothesis, affect it the corresponding translation
    * else, find the simplest translation allowing to provide l
  - Update the current hypothesis.

4.3 Simulation of the algorithm

Let us suppose that the first given example is : <John runs, run'(John')>.

- the syntactic hypotheses : as both words of p are unknown, we call A the category associated with *John* and B the one associated with *runs*. As p is a correct sentence, we know that A . B ——> S and we want to infer values for A and B. The only applicable rules in backward chaining are (L/) and (L\), as shown in Fig. 4 :

Fig. 4. the syntactic hypothesis

In these trees, sequents inside rectangles can only be axioms (because the category on the right side is a basic one) and there are ovals around sequents that could be developed further by decomposing the right side category and by applying rules (R/) or (R\), but which are cut at this level. So, the two possible solutions are :

* f(John)=A=U/V=S/B and f(runs)=B where B can be any category, basic or not;
* f(John)=A and f(runs)=B=V\U=A\S where A can be any category.

- the semantic translation :
  * Fig. 5 shows the translation of the first hypothesis

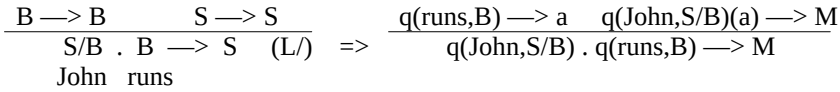


Fig. 5. translation of the first hypothesis

From M=q(John,S/B)(a)=q(John,S/B)(q(run,B))=run' (John') we infer that q(John,S/B)=run' and q(run,B)=John' . Another solution would be : q(run,A\S)=run' and q(John,S/B)=λP[P(John')] but we strictly stick to the simplest solution.

  * Fig. 6 shows the translation of the second hypothesis

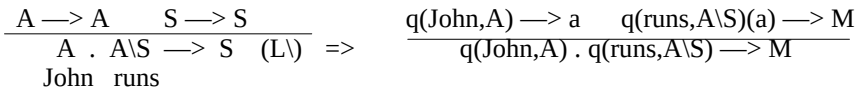


Fig. 6. translation of the second hypothesis

Similarly, we infer that q(runs,A\S)=run' and q(John,A)=John'.

At this stage, we have no reason to prefer one hypothesis to the other. The current hypothesis is then :

H={(John,S/B,run'),(runs,B,John')} OR {(John,A,John'),(runs,A\S,run')}.

Now, let us suppose that a second given example is <Paul runs, run' (Paul')>. The same process applies, except that *runs* now belongs to the current hypothesis.

- the syntactic hypotheses :

* if *runs* is assigned the category B, then *Paul* must receive S/B;
* if *runs* is assigned the category A\S, then *Paul* can receive either A or S/(A\S).

But the sequent A —> S/(A\S) is valid in the Lambek-Gentzen calculus (see rule R4), so S/(A\S) is in fact a particular case of A and is not to be taken into account.

- the semantic translation :

* the translation of the first tree, similar to the one of Fig. 6, gives rise to : M=q(Paul,S/B)(John')=run' (Paul') which is an equation without solution. So, the first subset of the current hypothesis is given up. It can be noticed that the hypothesis concerning *John* in this subset is also given up, although it was not concerned by the new example. A similar conclusion would have followed if the second example had been <John sleeps, sleeps' (John')>. Any other example sentence including one of the words concerned by the current hypothesis is enough to discard the wrong subset.

* the translation of the second tree, similar to the one of Fig. 7, gives rise to : M=run'(q(Paul,A))=run'(Paul') and allows to infer q(Paul,A)=Paul'.

The new current hypothesis is then :

H={(John,A,John'),(runs,A\S,run'),(Paul,A,Paul')}.

Without semantics, it would have been impossible to decide between the two initial possibilities. The only reason why *runs* must receive a fractional category is that its translation behaves like a function, a predicate.

In the appendix is given the analysis of <a man runs, ∃x[man' (x)∧run' (x)]> given as a new example. There are eleven syntactical possibilities, reduced to six really different ones. The semantic translation allows to abandon three of them and only four syntactico-semantic hypotheses are built, among which three will clearly be discarded very easily at the next repetition.

4.4 Treatment of polymorphism :

How can our algorithm assign several different categories to a unique word (as it seems necessary for determiners) ? A special module needs to be added. It becomes active when a new sentence example contains no new word but is impossible to analyze with the current hypothesis.

In this case, each word in the sentence, one after the other, should be treated as if it were the only one unknown so as to find out its other possible categories. Nevertheless, heuristics can help us chose the words more likely to be assigned several different categories :

- grammatical words, easy to recognize because they are the ones translated by lambda expressions (whereas lexical words are nearly always translated by a logical constant) are treated in priority.

- even if a word needs several categories, most of the time a unique logical translation is enough for all of them : this is true for determiners but also for pathological ambiguous words like *fly*, which can be either a common noun or a verb but is in both cases translated by *fly*'.

Further experiments, varying the order in which the examples are proposed, need to be performed to test if they are enough to treat every case of polymorphism.

5 Evaluation and conclusion

The algorithmic complexity of our algorithm is clearly exponential. More complex sentences would have entailed the multiplication of hypotheses and could have been

intractable. More precisely, our model seems to be particularly sensitive to the complexity of a new example *relatively to the current hypothesis*. This complexity can be measured by the number of new words appearing in an example. To master this complexity, we suggest to put *a priori* a bound on the number of new words appearing in a new sentence. An example which does not respect this bound is not treated but saved so as to be treated later, when the current hypothesis is developed enough. It is reasonable to think that children also learn from simple examples to more complex ones.

But our work also provides new insights on previous ones.

First, categorial grammars seem to be particularly adapted to the learning process. Recent research has found conditions under which the syntax of these grammars is learnable with positive examples ([3, 1, 12]). But, in these frameworks, only a simpler version of categorial grammars (restricted to rules R1 and R' 1) is considered. The main interest of L-categorial grammars is that they allow to restrict the number of different categories associated with each word. This number is a crucial parameter for the complexity of the syntactic analysis and for the treatment of polymorphism. Furthermore, in most previous work ([3, 12]), tree structures are provided as input data. In our model, thanks to the tree isomorphism, the semantics translation plays a similar role but in a weaker and more cognitively relevant fashion, as the functional structure of the logical formula gives indication about the functional syntactic structure of the sentence. Adriaans ([1]) also proposed a learning algorithm for categorial grammars using both syntax and semantics but he treated them separately : the semantic learning only started when the syntax learning was achieved instead of helping it as we propose.

Learning a natural language is certainly not equivalent to learning a formal grammar. *Natural* words and sentences *refer to things and situations* and can only be learned in their presence. Cognitive models built in this spirit ([10, 2, 14, 11]) already assumed this, but the syntaxes considered were more traditional and the semantic representations used were too close to syntactic structures ([21]) : they failed to represent complex logical relations like quantification or Boolean operators. Logical languages are more powerful and *a priori* independent from linguistic structures. Our model suggests that the acquisition of a conceptual representation of the world is necessary *before* the acquisition of the syntax of a natural language can start.

Fundamentally, what makes natural languages learnable in our model is the presupposition that *there exists an isomorphism between the syntax of sentences and their semantics*. This strong principle of compositionality is contested by linguists working on discourse phenomena but remains an interesting approximation. The *graph deformation condition* used in [2] was a weaker version of it.

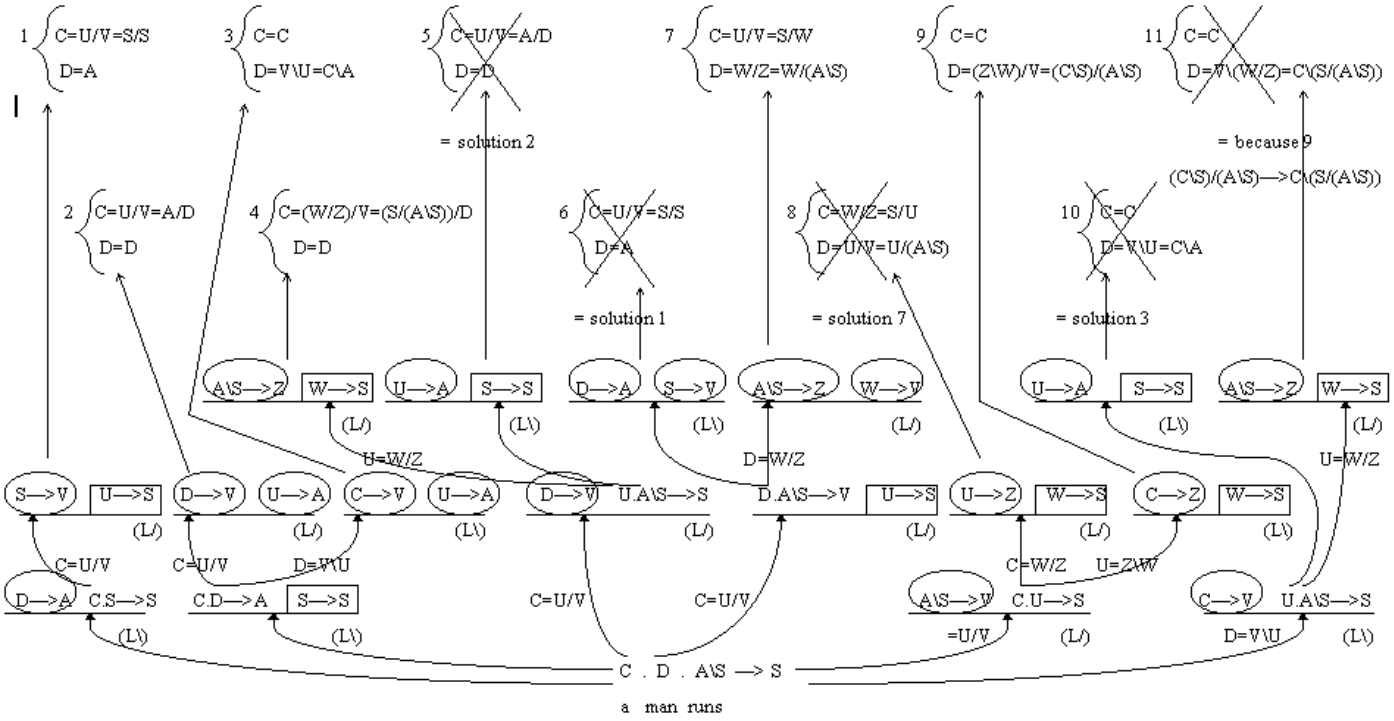
The work presented here is more a program of research than a full achievement, but seemed interesting enough to be exposed because our choices have theoretical, linguistic and cognitive motivations. The algorithm is being implemented and the search for a characterization of the languages it allows to learn is being explored.

References

1. P. W. Adriaans, *Language Learning from a Categorial Perspective*, Ph.D. thesis, University of Amsterdam, 1992.

2. J. R. Anderson, "Induction of Augmented Transition Networks", *Cognitive Science* 1, p125-157, 1977.
3. W. Buszkowski, G. Penn, "Categorial grammars determined from linguistic data by unification", *Studia Logica* 49, p431-454, 1990.
4. N. Chomsky, *Aspects of the Theory of Syntax*, Cambridge, MIT Press.
5. N. Chomsky, *Language and Mind*, Brace & World, 1968.
6. F. Denis, R. Gilleron, "PAC learning under helpful distributions", Proceedings of the 8th ACM Workshop on Computational Learning Theory, p132-145, 1997.
7. D. R. Dowty, R. E. Wall, S. Peters, *Introduction to Montague Semantics*, Reidel, Dordrecht, 1989.
8. A. Finkel, I. Tellier : "A polynomial algorithm for the membership problem with categorial grammars", *Theoretical Computer Science* 164, p207-221, 1996.
9. E. M. Gold, "Language Identification in the Limit", *Information and Control* 10, P447-474, 1967.
10. H. Hamburger, K. Wexler, "A mathematical Theory of Learning Transformational Grammar", *Journal of Mathematical Psychology* 12, p137-177, 1975.
11. J. C. Hill, "A computational model of language acquisition in the two-year-old", *Cognition and Brain Theory* 6(3), p287-317, 1983.
12. M. Kanazawa, "Identification in the Limit of Categorial Grammars", *Journal of Logic, Language & Information*, vol 5, n°2, p115-155, 1996.
13. J. Lambek, "The mathematics of Sentence Structure", *American Mathematical Monthly*, n°65, p154-170, 1958.
14. P. Langley, "Language acquisition through error discovery", *Cognition and Brain Theory* 5, p211-255, 1982.
15. M. Li, P. Vitanyi, "A theory of learning simple concepts under simple distributions", *SIAM J. Computing*, 20(5), p915-935, 1991.
16. R. Montague, *Formal Philosophy; Selected papers of Richard Montague*, Yale University Press, New Haven, 1974.
17. M. Moortgat, *Categorial investigations, logical and linguistic aspects of the Lambek Calculus*, Foris, Dordrecht, 1988.
18. R. T. Oehrle, E. Bach, D. Wheeler (eds.), *Categorial Grammars and Natural Language Structure*, Reidel, Dordrecht, 1988.
19. M. Pentus, "Lambek grammars are context-free", in : 8th Annual IEEE Symposium on Logic in Computer Science, Montreal, Canada, p429-433, 1992.
20. S. Pinker, "Formal models of language learning", *Cognition* 7, p217-283, 1979.
21. Y. Sakakibara, "Efficient learning of context-free grammars from positive structural examples", *Information & Computation* 97, p23-60, 1992.
22. S. Schieber, "Evidence against the context-freeness of natural languages", *Linguistics and Philosophy* 8, p333-343, 1995.
23. T. Shinohara, "Inductive inference of monotonic formal systems from positive data", p339-351 in : *Algorithmic Learning Theory*, S. Arikara, S. Goto, S. Ohsuga & T. Yokomori (eds), Tokyo : Ohmsha and New York and Berlin : Springer.
24. L. G. Valiant, "A theory of the learnable", *Communication of the ACM*, p1134-1142, 1984.
25. K. Wexler, P. Culicover, *Formal Principles of Language Acquisition*, Cambridge, MIT Press.
26. W. Zielonka, "Axiomatizability of Ajdukiewicz-Lambek calculus by means of cancellations schemes", *Zeischrift für Mathematsche Logik und grunlagen der Mathematik* 27, p215-224, 1981.

APPENDIX


1 C=U/V=S/S D=A
3 C=C D=V\U=C\A
5 C=U/V=A/D D=D
7 C=U/V=S/W D=W/Z=W/(A\S)
9 C=C D=(Z\W)/V=(C\S)/(A\S)
11 C=C D=V\(W/Z)=C\(S/(A\S))
= solution 2
= because 9
(C\S)/(A\S)—>C\(S/(A\S))
2 C=U/V=A/D D=D
4 C=(W/Z)/V=(S/(A\S))/D D=D
6 C=U/V=S/S D=A
8 C=W/Z=S/U D=U/W=U/(A\S)
10 C=C D=V\U=C\A
= solution 1
= solution 7
= solution 3
A\S—>Z W—>S
U—>A S—>S
D—>A S—>V
A\S—>Z W—>V
U—>A S—>S
A\S—>Z W—>S
(L/)
(L\)
U=W/Z
D=W/Z
S—>V U—>S
D—>V U—>A C—>V U—>A
D—>V U.A\S—>S
D.A\S—>V U—>S
U—>Z W—>S
C—>Z W—>S
C=U/V
C=U/V
D=V\U
D—>A C.S—>S
C.D—>A S—>S
C=U/V
C=U/V
C=W/Z
U=Z\W
A\S—>W C.U—>S
C—>V U.A\S—>S
=U/V
D=V\U
C . D . A\S —> S
a man runs

In this tree, unknown categories associated with *a* and *man* are noted C and D.

- semantic translation : let us call α=q(a, C) and β=q(man,D)

  * the translation of solution 1 leads to α(run'(β))=$\exists$x[man'(x)$\wedge$run'(x)].

The identification of both formulas imposes to define : β=x. But we forbid to introduce free variables into the logical translations of the words, so this solution is abandoned.

  * the translation of solution 2 leads to run'(α(β))=$\exists$x[man'(x)$\wedge$run'(x)].

This equation is impossible to solve, so this solution is abandoned.

  * the translation of solution 3 leads to run'(β(α))=$\exists$x[man'(x)$\wedge$run'(x)].

As in the previous case, this solution is abandoned.

  * the translation of solution 4 leads to (α(β))(run')=$\exists$x[man'(x)$\wedge$run'(x)].

To identify both formulas, we have to perform lambda-abstractions on the second one (they are only possible on logical *constants*) :

$\exists$x[man'(x)$\wedge$run'(x)]=λP$\exists$x[man'(x)$\wedge$P(x)](run') so α(β)=λP$\exists$x[man'(x)$\wedge$P(x)]

and λP$\exists$x[man'(x)$\wedge$P(x)]=λQλP$\exists$x[Q(x)$\wedge$P(x)](man')

so the simplest solution is : α=λQλP$\exists$x[Q(x)$\wedge$P(x)] and β=man'.

* the translation of solution 7 leads to α(β(run'))=$\exists$x[man'(x)$\wedge$run'(x)].

with $\exists$x[man'(x)$\wedge$run'(x)]=λP$\exists$x[man'(x)$\wedge$P(x)](run') we obtain two solutions :

- α=λx.x and β=λP$\exists$x[man'(x)$\wedge$P(x)]
- α=λP$\exists$x[man'(x)$\wedge$P(x)] and β=λx.x

(solutions using the constant function : λx.x are only considered in the cases where no other are possible, this is the reason why they were not proposed earlier).

* the translation of solution 9 leads to (β(run'))(α)=$\exists$x[man'(x)$\wedge$run'(x)].

After abstractions, the only solution left is : α=man' and β=λPλQ$\exists$x[Q(x)$\wedge$P(x)].

So, the hypotheses brought by this new example are :

{ (a, (S/(A\S))/D, λQλP$\exists$x[Q(x)$\wedge$P(x)]), (man, D, man') }

OR { (a, S/W, λx.x), (man, W/(A\S), λP$\exists$x[man'(x)$\wedge$P(x)]) }

OR { (a, S/W, λP$\exists$x[man'(x)$\wedge$P(x)])), (man, W/(A\S), λx.x) }

OR { (a, C, man'), (man, (C\S)/(A\S), λPλQ$\exists$x[Q(x)$\wedge$P(x)]) }.

It is obvious that only the first subset will resist to new examples using one of these words.