

On Translating Context-Free Grammars into Lambek Categorical Grammars

Stepan Kuznetsov

Steklov Mathematical Institute, Russian Academy of Sciences
sk@mi.ras.ru

Abstract

We modify Buszkowski's translation of context-free grammars to Lambek categorical grammars in order to deal with Montague-style semantics properly.

1 Types, Terms, and Lambek Categorical Grammars

Introduce the notion of *semantic type* in the following way. Let $\mathbf{B}$ be a (countable) set of *basic types*. *Semantic types* are built from basic types using one operation “ $\rightarrow$ ”. We denote the set of all semantic types by $\mathbf{T}$. Types are populated with *terms* (also called λ -terms): for every semantic type there is a countable number of variables of that type; we can also allow constants—they behave just like variables, but for them substitution and λ -abstraction is prohibited. If f is of type $A \rightarrow B$ and u is of type A , then (fu) is of type B ; if v is of type B and x is a variable of type A , then $\lambda x.v$ is of type $A \rightarrow B$. If term u is of type A , then we write “ $u : A$ ”. By $u[x := v]$ we denote the result of substituting term v for all free (not under λ) occurrences of variable x . The set of all terms will be denoted by Tm_λ .

Terms are considered modulo α -, β - and η -equivalence: one can freely rename bound variables and perform reductions on subterms: $(\lambda x.u)v \rightsquigarrow_\beta u[x := v]$ (if free variables of v are not bound in u) and $(\lambda x.(ux)) \rightsquigarrow_\eta u$, if x is not a free variable of u .

On the other hand, we also have *syntactic types*. They are built from a set of *primitive types* Pr using two binary connectives $\backslash$ and $/$ (left and right division). The set of all syntactic types is denoted by Tp (more precisely, $\text{Tp}(\backslash, /)$).

We introduce a translation $\sigma : \text{Tp} \rightarrow \mathbf{T}$ of syntactic types to semantic ones. On Pr this function is defined arbitrarily (the image of a primitive syntactic type need not be a basic type), and then σ is uniquely propagated to Tp : $\sigma(A \backslash B) = \sigma(B / A) = \sigma(A) \rightarrow \sigma(B)$.

We denote both syntactic and semantic types by capital Latin letters ($A, B, C, \dots$). If A is a syntactic type and u is a term, we can also write “ $u : A$ ”, meaning $u : \sigma(A)$.

Now we introduce the product-free Lambek calculus. This calculus originates from [7], but here we enrich it with terms attached to types, in a Curry – Howard style (due to van Benthem [1]; see, for example, [3]). Capital Greek letters denote sequences of syntactic types. If $\Pi = A_1 \dots A_n$ and $\vec{x} = (x_1, \dots, x_n)$ is a sequence of variables, then by $\vec{x} : \Pi$ we denote the sequence $x_1 : A_1, \dots, x_n : A_n$. Expressions of the form $\vec{x} : \Pi \rightarrow u : A$ are called *sequents*. Axioms are sequents of the form $x : A \rightarrow x : A$. The rules are as follows:

$$\frac{x : A, \vec{y} : \Pi \rightarrow u : B}{\vec{y} : \Pi \rightarrow (\lambda x.u) : (A \backslash B)} \quad \frac{\vec{y} : \Pi, x : A \rightarrow u : B}{\vec{y} : \Pi \rightarrow (\lambda x.u) : (B / A)} \text{ where } \Pi \text{ is non-empty}$$

$$\frac{\vec{x} : \Pi \rightarrow u : A \quad \vec{y} : \Gamma, t : B, \vec{z} : \Delta \rightarrow v : C}{\vec{y} : \Gamma, \vec{x} : \Pi, f : (A \backslash B), \vec{z} : \Delta \rightarrow v[t := (fu)] : C}$$

$$\frac{\vec{x} : \Pi \rightarrow u : A \quad \vec{y} : \Gamma, t : B, \vec{z} : \Delta \rightarrow v : C}{\vec{y} : \Gamma, f : (B / A), \vec{x} : \Pi, \vec{z} : \Delta \rightarrow v[t := (fu)] : C}$$

Let us call this calculus L_λ . Note that if $L_\lambda \vdash \vec{x} : \Pi \rightarrow u : A$, then all free variables of u are contained in $\vec{x}$, and indeed u is of type A .

Finally, the cut rule is admissible and acts as the substitution operation on terms:

$$\frac{\vec{x} : \Pi \rightarrow u : A \quad \vec{y} : \Gamma, t : A, \vec{z} : \Delta \rightarrow v : C}{\vec{y} : \Gamma, \vec{x} : \Pi, \vec{z} : \Delta \rightarrow v[t := u] : C}$$

Now define the notion of L_λ -grammar (semantically enriched Lambek categorial grammar): it is a tuple $\mathcal{G} = \langle \Sigma, \mathcal{D}, H \rangle$. Here Σ is a finite alphabet, $H \in \text{Tp}$ is the final type, and $\mathcal{D} \subset \Sigma \times \text{Tp} \times \text{Tm}_\lambda$ is the *categorial dictionary*. The set $\mathcal{D}$ is finite, and if $\langle a, A, u \rangle \in \mathcal{D}$, then $u : A$. Implicitly, the mapping $\sigma : \text{Tp} \rightarrow \mathbf{T}$ is also part of $\mathcal{G}$.

A string $w = a_1 \dots a_n$ is accepted by this grammar iff there exist such syntactic types $A_1, \dots, A_n$ and terms $u_1, \dots, u_n$, such that $\langle a_i, A_i, u_i \rangle \in \mathcal{D}$ ($i = 1, \dots, n$), and $L_\lambda \vdash x_1 : A_1, \dots, x_n : A_n \rightarrow v : H$ (for some term v). The term $v[x_1 := u_1, \dots, x_n := u_n]$ is called the *semantic value* of w w.r.t. $\mathcal{G}$. Note that a grammar can associate several semantic values to the same w via various choices of A_i or various derivations of the sequent.

2 Context-Free Grammars with Bidirectional Labelling

Here we consider only context-free grammars in Chomsky normal form [4]. Such a grammar is a tuple $G = \langle N, \Sigma, \mathcal{P}, s \rangle$, where N and Σ are disjoint alphabets (elements of N are called non-terminal symbols), $s \in N$ is the starting symbol, and $\mathcal{P}$ is the set of rules; each rule is either of the form $p \Rightarrow qr$, where $p, q, r \in N$, or of the form $p \Rightarrow a$, where $p \in N$, $a \in \Sigma$. A rule $p \Rightarrow \alpha$ can be applied to a word $\eta p \theta \in (N \cup \Sigma)^+$, producing $\eta \alpha \theta$. If $w \in \Sigma^+$ can be obtained from s by several applications of the rules, then w is generated by grammar G .

Now we introduce semantic labels (see, for example, [5]). Firstly, we associate a semantic type with any non-terminal symbol. This is done by a function $\sigma : N \rightarrow \mathbf{T}$. For any rule of the form $p \Rightarrow qr$ we require that either $\sigma(r) = \sigma(q) \rightarrow \sigma(p)$ or $\sigma(q) = \sigma(r) \rightarrow \sigma(p)$. In the first case we call $p \Rightarrow qr$ a right-to-left rule, in the second case—a left-to-right one. Finally, with any rule of the form $p \Rightarrow a$ we associate a term u of type $\sigma(p)$.

Let $w \in \Sigma^+$ be generated by G in the following way $s \Rightarrow \alpha_1 \Rightarrow \dots \Rightarrow \alpha_m \Rightarrow w$ (each step here is an application of one rule). Going from the end to the beginning of this derivation, we associate a term with each occurrence of each non-terminal symbol. In the case of $p \Rightarrow a$ we take u ; in the case of a right-to-left rule $p \Rightarrow qr$, if u is associated with r and v is associated with q , then for p we take (uv) . Note that, since $u : \sigma(q) \rightarrow \sigma(p)$, and $v : \sigma(q)$, (uv) is a well-formed term of type $\sigma(p)$. The left-to-right rule case is symmetric (we take (vu) ; $v : \sigma(r) \rightarrow \sigma(p)$, $u : \sigma(r)$).

The term associated with the initial s is considered to be the semantic value of w .

3 Translation of Context-Free Grammars into Lambek Categorial Grammars

Given a context-free grammar G with semantic labels, we aim to obtain an L_λ -grammar $\mathcal{G}$ that generates the same set of words, and, moreover, gives the same semantic values to words of this set as G does. Essentially we use Buszkowski's [2] construction, decorated with semantic labels.

Consider all non-terminal symbols as primitive types ($N \subseteq \text{Pr}$); the σ mapping is also preserved. With any $t \in N$ we associate a set $I(t) \subset \text{Tp}$ in the following way:

- $t \in I(t)$;
- if $p \Rightarrow qt$ is a rule of G , then $(q \setminus p) \in I(t)$;
- if $t \in N$ and $p \Rightarrow qr$ is a rule of G , then $(q \setminus p) / (t \setminus r) \in I(t)$.

Now we are ready to define $\mathcal{G}$. Let $H = s$. For any rule of the form $t \Rightarrow a$ and $A \in I(t)$ with a term $u : \sigma(t)$ associated we add an entry to the categorial dictionary $\mathcal{D}$ in the following way.

- If $A = t$, then we add $\langle a, A, u \rangle$.
- If $A = q \setminus p$ and $p \Rightarrow qt$ is a right-to-left rule, then $\sigma(t) = \sigma(q) \rightarrow \sigma(p)$, so $u : A$, and we add $\langle a, A, u \rangle$.
- If $A = q \setminus p$ and $p \Rightarrow qt$ is a left-to-right rule, then $\sigma(q) = \sigma(t) \rightarrow \sigma(p)$. Let f be a variable of type $\sigma(q)$ and let $\tilde{u} = \lambda f.(fu)$. Then $\tilde{u}$ is a well-formed term of type $\sigma(q) \rightarrow \sigma(p) = \sigma(q \setminus p)$, and we add $\langle a, A, \tilde{u} \rangle$.
- If $A = (q \setminus p) / (t \setminus r)$ and $p \rightarrow qr$ is a right-to-left rule, then, since $\sigma(r) = \sigma(q) \rightarrow \sigma(p)$, $\sigma(A) = (\sigma(t) \rightarrow \sigma(r)) \rightarrow \sigma(r)$. Let g be a variable of type $(\sigma(t) \rightarrow \sigma(r))$ and let $\tilde{u} = \lambda g.(gu)$. We have $\tilde{u} : ((\sigma(t) \rightarrow \sigma(r)) \rightarrow \sigma(r))$, whence $\tilde{u} : A$, and we add $\langle a, A, \tilde{u} \rangle$.
- If $A = (q \setminus p) / (t \setminus r)$ and $p \rightarrow qr$ is a left-to-right rule, then $\sigma(q) = \sigma(r) \rightarrow \sigma(p)$. Thus $\sigma(A) = (\sigma(t) \rightarrow \sigma(r)) \rightarrow ((\sigma(r) \rightarrow \sigma(p)) \rightarrow \sigma(p))$. Let g be a variable of type $(\sigma(t) \rightarrow \sigma(r))$ and f be a variable of type $(\sigma(r) \rightarrow \sigma(p))$. Then $\hat{u} = \lambda g.\lambda f.(f(gu))$ is a well-formed term of type $\sigma(A)$, and we add $\langle a, A, \hat{u} \rangle$ to $\mathcal{D}$.

Proposition 1. *If $w \in \Sigma^+$ is accepted by $\mathcal{G}$, then it is also generated by G .*

Proof. For every right-to-left rule of the form $p \Rightarrow qr$ consider the sequent $x : q, f : r \rightarrow (fx) : p$, and for every left-to-right rule $p \Rightarrow rq$ consider the sequent $f : r, x : q \rightarrow (fx) : p$. These sequents form a set $\mathcal{A}$. Let's add them as new axioms and also add the cut rule, thus getting a new calculus $L_\lambda + \mathcal{A}$.

Now let $t \in N$ and $A \in I(t)$. Then one can easily prove that $L_\lambda + \mathcal{A} \vdash x : t \rightarrow x' : A$, where x' is a term of type $\sigma(A)$ (actually, either x , or $\tilde{x}$, or $\hat{x}$).

Now, if $w = a_1 \dots a_n$ is accepted by $\mathcal{G}$, then $L_\lambda \vdash x_1 : A_1, \dots, x_n : A_n \rightarrow v : s$. Using the cut rule, we get

$$L_\lambda + \mathcal{A} \vdash x_1 : t_1, \dots, x_n : t_n \rightarrow v[x_1 := x'_1, \dots, x_n := x'_n] : s.$$

Now we reformulate the calculus $L_\lambda + \mathcal{A}$ in an equivalent form. Instead of the set $\mathcal{A}$ of axioms we introduce a set $\mathcal{R}$ of rules:

$$\frac{\vec{z}_1 : \Gamma_1 \rightarrow v : q \quad \vec{z}_2 : \Gamma_2 \rightarrow u : r}{\vec{z}_1 : \Gamma_1, \vec{z}_2 : \Gamma_2 \rightarrow (uv) : p} \text{ for a right-to-left rule } p \Rightarrow qr;$$

$$\frac{\vec{z}_1 : \Gamma_1 \rightarrow u : r \quad \vec{z}_2 : \Gamma_2 \rightarrow v : q}{\vec{z}_1 : \Gamma_1, \vec{z}_2 : \Gamma_2 \rightarrow (uv) : p} \text{ for a left-to-right rule } p \Rightarrow rq.$$

The calculi $L_\lambda + \mathcal{A}$ and $L_\lambda + \mathcal{R}$ are equivalent. Moreover, $L_\lambda + \mathcal{R}$ enjoys the cut-elimination property. Therefore,

$$L_\lambda + \mathcal{R} \vdash x_1 : t_1, \dots, x_n : t_n \rightarrow v' : s$$

without using the cut rule. Since types here do not contain any connectives, the only rules that can be used in the derivation are rules from $\mathcal{R}$, and the derivation itself is actually a context-free derivation in G , thus w is generated by G . $\square$

Lemma 1 shows that $\mathcal{G}$ does not accept any extra words in comparison with G . Moreover, the cut-elimination procedure in $L_\lambda + \mathcal{R}$ preserves the semantic value (modulo α -, β - and η -equivalence; this can be proved in the same way as for L_λ , see [6]). Therefore v' is the same term as $v[x_1 := x'_1, \dots, x_n := x'_n]$. Finally, on one hand, v' is the semantic value assigned by G . On the other hand, $\mathcal{G}$ assigns the term $v[x_1 := x'_1, \dots, x_n := x'_n]$, where the choice of x_i , $\hat{x}_i$ or $\tilde{x}_i$ for x'_i is determined by A_i . Thus the semantic values coincide.

Proposition 2. *If $w \in \Sigma^+$ is generated by G , then w is accepted by $\mathcal{G}$ with the same semantic value.*

Proof. Proceed by induction on G -derivation. Decorate all non-terminal symbols with syntactic types. The start symbol s is decorated with itself. On the induction step we apply a rule $p \Rightarrow qr$, decorate r with $(q \setminus p)$ and consider several cases.

- p is decorated with itself. Then decorate q with q .
- p is decorated with a type of the form $(q_1 \setminus p_1)$. Then decorate q with $(q_1 \setminus p_1) / (q \setminus p)$.
- p is decorated with a type of the form $(q_1 \setminus p_1) / (p \setminus s)$. Then decorate q with $(q_1 \setminus p_1) / (q \setminus s)$.

One can prove by induction that any occurrence of a non-terminal symbol t is decorated with a type from $I(t)$.

Now we backtrack the G -derivation and associate terms to occurrences of non-terminal symbols. This correspondence is actually twofold: one line of terms comes from the G -derivation itself, the other is going to come from the Lambek derivation. We shall prove the following statement by induction: if at some step of the G -derivation starting with s we get $t_1 \dots t_m$ ($t_i \in N$), and for each t_i we have a term u_i associated, then there exist terms $u'_1, \dots, u'_m$, such that for every i either $u'_i = u_i$, or $u'_i = \tilde{u}_i$, or $u'_i = \hat{u}_i$, so that $u'_i : A_i$,

$$L_\lambda \vdash x_1 : A_1, \dots, x_m : A_m \rightarrow \bar{v} : s,$$

and $\bar{v}[x_1 := u'_1, \dots, x_m := u'_m] = v$ is the term associated with the start symbol s (the semantic value); the syntactic types A_i are those with which t_i were decorated.

The base case is trivial: $m = 1$, $A_1 = s$, and we just take $u'_1 = v$.

The step with the right-to-left rule is considered as follows. Let the rule be of the form $p \Rightarrow qr$, and the term u associated with p is then of the form (hu_1) , h associated with r , u_1 associated with q . Also let p be decorated with A , q with B and r with C . We check that

$$L_\lambda \vdash u'_1 : B, h' : C \rightarrow u' : A,$$

by considering possible cases for u'_1 , h' , and u' , and then apply the cut rule. The left-to-right case is considered symmetrically.

Finally, we apply the rules of the form $t_i \Rightarrow a_i$. Since the choice of the alternative for u'_i is determined by A_i , the corresponding entries $\langle a_i, A_i, u'_i \rangle$ exist in $\mathcal{D}$. Therefore the word $a_1 \dots a_n$ is accepted by $\mathcal{G}$ with semantic value v , q.e.d. $\square$

We've shown that the translation of context-free grammars to Lambek categorial grammars not only keeps the language as a set of words, but also preserves the semantic values given by the context-free derivation.

The vice versa direction (from Lambek grammars to context-free ones) is potentially more interesting. As proved by Pentus [8], every language generated by a Lambek categorial grammar is context-free. However, being rather straightforward, Pentus' translation is quite inefficient (leads to exponential growth of the grammar). The question of the behaviour of semantic values under this translation remains open.

Acknowledgments

The author is grateful to Prof. Mati Pentus for fruitful discussions. This research was supported by the Russian Foundation for Basic Research (grants 12-01-00888-a and 14-01-00127-a) and by the Presidential Council for Support of Leading Scientific Schools (grant NŠ-1423.2014.1).

References

- [1] J. van Benthem. The semantics of variety in categorial grammar. *Report 83-29*, Dept. of Mathematics, Simon Fraser University, Vancouver, 1983.
- [2] W. Buszkowski. On the Equivalence of Lambek Categorical Grammars and Basic Categorical Grammars. — Amsterdam: Institute for Logic, Language and Computation, University of Amsterdam, 1993. — 21 p. — (ILLC Prepublication Series; LP-93-07).
- [3] B. Carpenter. Type-logical semantics. — Cambridge, Mass.: The MIT Press, 1997.
- [4] N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*. **I T-2**, No. 3, 1956. — P. 113–124.
- [5] D. R. Dowty et al. Introduction to Montague semantics. — Dordrecht etc: D. Reidel, 1981.
- [6] H. Hendriks. Studied flexibility: categories and types in syntax and semantics. Ph. D. dissertation, University of Amsterdam, 1993.
- [7] J. Lambek. The mathematics of sentence structure. *Amer. Math. Monthly*. **65**, No. 3, 1958. — P. 154–170.
- [8] M. Pentus. Lambek grammars are context-free. *Proc. of the 8th Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Society Press, Los Alamitos, California, 1993. — P. 429–433.