Cobordisms and commutative categorial grammars

Sergey Slavnov
National Research University Higher School of Economics
sslavnov@yandex.ru

Abstract

We propose a concrete surface representation of abstract categorial grammars in the category of word cobordisms or cowardisms for short, which are certain bipartite graphs decorated with words in a given alphabet, generalizing linear logic proof-nets. We also introduce and study linear logic grammars, directly based on cobordisms and using classical multiplicative linear logic as a typing system.

1 Introduction

The best known categorial grammars are based on noncommutative variants of linear logic, most notably, on Lambek calculus [Lam58] and its variations/extensions. On the other hand, such formalisms as abstract categorial grammars (ACG) [dG01], also known (with minor variations) as λ-grammars [Mus07] or linear grammars [MP12], arise from an alternative or, rather, complementary approach, and use ordinary implicational linear logic and linear λ-calculus. These can be called “commutative” in contrast to the “noncommutative” Lambek grammars. Commutative grammars are attractive because of the much more familiar and intuitive underlying logic, besides they are remarkably expressive. Unfortunately, basic constituents of ACG used for syntax generation seem extremely abstract: they are just linear λ-terms. Identifying an abstract λ-term with some element of language is not so easy, and syntactic analysis becomes complicated. It seems that some more concrete surface representation for commutative grammars would be highly desirable.

In this work we propose that such a representation is indeed possible. We introduce a specific structure of word cobordisms, or, simply cowardisms, as we abbreviate for a joke.

Word cobordism is a bipartite graph, more precisely, a perfect matching (generalizing linear logic proof-nets), whose edges are labeled with words in a given alphabet, and whose vertex set is subdivided into the input and the output parts. This can be seen as a one-dimensional topological cobordism (see [Sto16], [BD95]) decorated with words, which explains our terminology. (For
a pedestrian discussion of cobordisms that might be relevant to the content of this paper, see [BS11].

Just as topological cobordisms, word cobordisms can be organized into a category, with composition given by gluing inputs to outputs. The resulting category has a rich structure, in particular it is compact closed (see [KL80], also [AC09]), and, as any compact closed category, it provides a denotational model for multiplicative linear logic and for linear $\lambda$-calculus. The latter model gives rise to the geometric cowordism representation of string ACG that we discuss.

On the other hand, the very structure of cowordism category with its involutive duality, suggests using classical (rather than intuitionistic) multiplicative linear logic ($\text{MLL}$) as more natural for this setting. Thus, we also define and study linear logic grammars (LLG), based directly on the cowordism representation and using $\text{MLL}$ as the typing system. String ACG can be seen as a particular case of LLG.

LLG with their underlying compact category could be seen as a commutative version of pregroup grammars (see [Lam99]). This suggests possible connections with categorical compositional distributional semantics (DisCoCat) [CSC10], which use pregroup grammars a lot. Indeed, DisCoCat models are based on finite-dimensional vector spaces and use their symmetric and compact closed categorical structure in an essential way. Arguably, LLG match these structures just better than other syntactic formalisms. Although such a matching is not required for any known construction, and we cannot say even if it is useful at all, it seems at least interesting that a parallel symmetric compact structure can be found on the syntactic side as well.

(We should add though that the above parallelism does not go to the extreme. Typically, DisCoCat models, apart from using the canonical symmetric compact structure of vector spaces, impose additional, non-canonical structure of commutative Frobenius algebra, so called “spiders” [CSC13]. This latter commutativity has no analogue on the syntactic, surface level.)

In any case, we think that the cowordism representation with its simple geometric meaning and diagrammatic reasoning might be helpful for studying language generation, some examples are given. Hopefully, it can also be used for applying ideas of DisCoCat to various “commutative” formalisms, thus going beyond context-free languages.

2 Boundaries and multiwords

Let $T$ be a finite alphabet. We denote the set of all finite words in $T$ as $T^*$, and the empty word as $\epsilon$. For consistency of definitions, we will also need cyclic words, which are equivalence classes of elements of $T^*$ quotiented by cyclic permutations of letters. For $w \in T^*$ we denote the corresponding cyclic word as $[w]$.

For a set $X$ of natural numbers and an integer $n$ we use the notation $n + X = \{n + m | m \in X\}$, and $n - X = \{n - m | m \in X\}$. For multisets $A, B$ we denote their disjoint union as $A + B$. For a positive integer $N$, we denote
A multiword is defined by $I(1) = 0$, $I(0) = \emptyset$. Given two boundaries $X$ and $Y$, the tensor product boundary $X \otimes Y$ and the dual boundary $X^\perp$ are obtained, respectively, by concatenation and order and polarity reversal, i.e. $|X \otimes Y| = |X| + |Y|$, $(X \otimes Y)^⊥ = X_0 \cup (|X| + Y_1)$, and $|X^⊥| = |X|$, $(X^⊥)_r = |X| + 1 - X_r$. The neutral element for tensor product is the unit boundary 1 defined by $I(1) = 0$, $I(0) = \emptyset$.

Note that we have the identity $(X \otimes Y)^⊥ = Y^⊥ \otimes X^⊥$. This should not suggest any sort of noncommutativity in the category of boundaries. We will have a natural isomorphism between $(X \otimes Y)^⊥$ and $X^⊥ \otimes Y^⊥$, just not equality. The flip of tensor factors will allow somewhat better pictures, with fewer crossings.

Given an alphabet $T$, a regular multiword $M$ over $T$ with boundary $X$ is a directed graph on the set $I(X)$ of vertices, whose edges are labelled with words in $T^*$, such that each vertex is adjacent to exactly one edge (so that it is a perfect matching), and for every edge its left endpoint is in $X_l$ and its right endpoint is in $X_r$.

In the following we will identify a regular multiword with the set of its labeled edges. The notation $[x, w, y]$ will stand for an edge from $x$ to $y$ labeled with the word $w$.

A general multiword $M$ over $T$ with boundary $X$ is defined as a pair $M = (M_0, M_c)$, where $M_0$, the regular part, is a regular multiword over $T$ with the boundary $X$, and $M_c$, the singular part, is a finite multiset of cyclic words over $T$. A multiword is acyclic or regular if its singular part is empty. Otherwise it is singular.

Singular multiwords should be understood as pathological (in the context of this work), but we need them for consistency of definitions. Geometrically, a multiword can be understood as the disjoint union of an edge-labeled graph and a collection of closed curves (i.e. circles) labeled with cyclic words.
We will use certain conventions for depicting multiwords, which guarantee unambiguous reading of pictures. Unless otherwise stated, points of the boundary are ordered from left to right. Left endpoints are marked as solid dots, and right endpoints as arrowheads. Also, our strict convention for reading edge labels is that words in a picture are always read from left to right, in the usual way, no matter what is the direction of edges. An example is in Figure 1.

Given two multiwords $M = (M_0, M_e)$ and $N = (N_0, N_c)$ with boundaries $X$ and $Y$ respectively, the tensor product multiword $M \otimes N$ has boundary $X \otimes Y$ and is defined as the disjoint union, i.e. $(M \otimes N)_c = M_c + N_c$ and $(M \otimes N)_0 = \{[i, w, j] \mid [i, w, j] \in M_0\} \cup \{([X] + i, w, [X] + j) \mid [i, w, j] \in N_0\}$.

A crucial operation on multiwords is contraction, which consists in gluing neighboring endpoints of opposite polarity and concatenating the corresponding edge labels in the direction from the left endpoint to the right. Here is an accurate definition.

Let $M$ be a multiword with boundary $X$, and $n < |X|$ be such that $n$ and $n + 1$ have opposite polarity in $X$. Let $x$ be the right endpoint in the pair $(n, n + 1)$ and $y$ be the left one.

The elementary contraction $\langle M \rangle_{n,n+1}$ of $M$ along $n$ and $n + 1$ is the multiword $M'$ with the boundary $X'$, where $|X'| = |X| - 2$, $\langle X' \rangle_l = (\text{shift}^2_0)^{-1}(X_l)$, constructed as follows.

If $x$ and $y$ are not connected with an edge in $M_0$, then $M'_e = M_e$, and $M'_0$ consists of all edges $[i, w, j]$ such that $[\text{shift}^2_0(i), w, \text{shift}^2_0(j)] \in M_0$ plus the edge $[\alpha, uv, \beta]$ such that $[\text{shift}^2_0(\alpha), u, x]$, $[y, v, \text{shift}^2_0(\beta)]$ are in $M_0$.

If there is an edge $[y, w, x] \in M_0$, then $M'_e = M_e + \{[w]\}$, and $M'_0$ consists of all edges $[i, w, j]$ such that $[\text{shift}^2_0(i), w, \text{shift}^2_0(j)] \in M_0$.

It is easy to see that in all cases $M'_0$ is a perfect matching and its edges start at left endpoints of $X'_l$. Also, when the contracted vertices $x$ and $y$ happen to be connected with an edge, the resulting multiword necessarily is singular.

Elementary contractions can be iterated.

Let $X, Y$ be boundaries, $i \in N$ and assume that $i + Y^\perp \otimes Y$ is a subboundary of $X$. Let $n = |Y| = |Y^\perp|$. Then for any multiword $M$ with boundary $X$ we define the iterated contraction $\langle M \rangle_{i+Y^\perp \otimes Y}$ of $M$ along $i + Y^\perp \otimes Y$ by $\langle M \rangle_{i+Y^\perp \otimes Y} = (\ldots (\langle \langle M \rangle_{i+n,i+n+1}\rangle_{i+n-1,i+n})\ldots)_{i+1,i+2}$.

It is easy to check that the above is well defined.
In order to avoid possible ambiguity in pictures with iterated contractions, we will use quotation marks. An example is shown in Figure 2, where dotted lines connect neighboring vertices that will be contracted. When we replace dotted lines with solid ones, the resulting graph has discontinuous edge labels, and it is not immediately clear how to read them. Our convention is that any block in quotation marks is read from left to right, as usual, while several blocks labeling one edge are read in the order in which they appear as we traverse the edge from the left endpoint to the right one. In particular, in Figure 2, when all the zigzagging is reduced, we obtain the sentence “Jim goes out with Ann a lot”.

3 Word cobordisms

Let $X, Y$ be boundaries and $T$ be an alphabet.

A word cobordism or, simply, a cowordism $\sigma : X \to Y$ over $T$ from $X$ to $Y$ is a multiword over $T$ with boundary $X^\perp \otimes Y$. We say that $Y$ is the outgoing boundary of $\sigma$, and $X$ is the incoming boundary. A cowordism is regular if its underlying multiword is regular, otherwise it is singular.

When depicting a cowordism $\sigma : X \to Y$, we put elements $1, \ldots, |X|$ of the subboundary $X^\perp$ on one vertical line, with the increasing order corresponding to the direction up, and we put the elements $|X| + 1, \ldots, |X| + |Y|$ of the subboundary $|X| + Y$ on a parallel line to the right, in the increasing order corresponding to the direction down.

For example, if the boundaries $X, Y$ are given by

$$|X| = 4, \quad X_l = \{3\}, \quad |Y| = 4, \quad Y_l = \{2\},$$

then a cowordism $\sigma : X \to Y$ will be depicted as in Figure 3a (where we indicate vertex numbers for clarity). The subboundary $X^\perp$ of $\sigma$ corresponds to the incoming boundary $X$ by means of an order and polarity reversing bijection. In particular the right endpoint 2 in the picture corresponds to the left endpoint 3 of $X$. 
In general, when the structure of boundaries is not important, we “squeeze” parallel edges into one and represent a cowordism $\sigma : X \rightarrow Y$ schematically as a box with an incoming wire labeled with $X$ and an outgoing wire labeled with $Y$. More generally, we represent a cowordism $\sigma : X_1 \otimes \ldots \otimes X_n \rightarrow Y_1 \otimes \ldots \otimes Y_m$ as a box whose $n$ incoming wires are labeled with $X_i$’s and $m$ outgoing wires are labeled with $Y_i$’s, as in Figure 3b. Such a “squeezed” picture is consistent with the full picture. If we “expand” each edge into parallel edges adjacent to points in the corresponding subboundary, we obtain the detailed picture. When we depict a cowordism $\sigma : 1 \rightarrow X$, respectively $\tau : X \rightarrow 1$, we do not have wires on the left, respectively right.

This is, of course, a variation of the familiar pictorial language for monoidal categories. Note, however, that, since cowordisms are, by definition, geometric objects, the diagrammatic representation is quite literal, and diagrammatic reasoning is valid automatically, without further justification.

Matching cowordisms are composed by gluing incoming and outgoing boundaries.

Let boundaries $X, Y, Z$ and cowordisms $\sigma : X \rightarrow Y$, $\tau : Y \rightarrow Z$, with the underlying multiwords $M_\sigma$, $M_\tau$ respectively be given. The composition $\tau \circ \sigma : X \rightarrow Z$ is the cowordism whose underlying multiword $M_{\tau \circ \sigma}$ is obtained as the iterated contraction $M_{\tau \circ \sigma} = \langle M_\sigma \otimes M_\tau \rangle_{|X|+Y|Y|}$. It is easy to see that, with our conventions, composition of cowordisms $\sigma : X \rightarrow Y$, $\tau : Y \rightarrow Z$ corresponds to the schematic picture in Figure 3c.

We get a detailed, “full” picture by expanding every edge into as many parallel edges as there are points in the corresponding boundary. For example, if $X, Y$ are as in (1), and $Z$, say, has two points of opposite polarity, then the schematic picture in Figure 3c translates to the detailed picture in Figure 3d.

It is evident from geometric representation that composition of cowordisms is associative.

The identity cowordism $\text{id}_X : X \rightarrow X$ is the regular multiword with the boundary $X^\perp \otimes X$ defined as

$$\text{id}_X = \{[[|X| \pm i, \epsilon, |X| - i + 1]] \mid i \in X_l\} \cup \{[[|X| - i + 1, \epsilon, |X| + i]] \mid i \in X_r\}.$$  

In a schematic, “squeezed” picture, the identity cowordism corresponds to a single wire: $\text{id}_X : X \longrightarrow X$. In the full picture there are as many parallel wires as there are points in $X$. If $X$ is as in (1), then the full picture is the following: $\text{id}_X : X \longrightarrow X$.

Now let boundaries $X, Y, Z, T$ and cowordisms $\sigma : X \rightarrow Y$, $\tau : Z \rightarrow T$ be given. Let us write $\sigma_0$, respectively $\tau_0$, for the regular part of (the underlying multiword of) $\sigma$, respectively $\tau$, and let us write $\sigma_c$, $\tau_c$ for the respective singular parts.

The tensor product cowordism $\sigma \otimes \tau : X \otimes Z \rightarrow Y \otimes T$ is defined by the multiword with the singular part $(\sigma \otimes \tau)_c = \sigma_c + \tau_c$, and the regular part $(\sigma \otimes \tau)_0$ obtained as the union of edge sets $\sigma_0$, $\tau_0$ appropriately shifted:

$$(\sigma \otimes \tau)_0 = \{[i + |Z|, w, j + |Z|] \mid i, w, j \in \sigma_0\} \cup$$
In the graphical language, tensor product of cowordisms corresponds simply to putting two boxes side by side, as in Figure 4a.

The \textit{symmetry cowordism} $s_{X,Y} : X \otimes Y \to Y \otimes X$ is defined by the regular multiword with the set of edges
\[
\{ [[|Y|+|X|]i, w, |X|+|Y|+i] \mid i \in Y_r \} \cup \{ [[|X|+|Y|+i, \epsilon, |Y|+|X|+i] \mid i \in X_r \} \cup \{ [[|X|+|Y|+i, \epsilon, |Y|+|X|+i] \mid i \in X_l \} \cup \{ [[|X|+|Y|+i, \epsilon, |Y|+|X|+i] \mid i \in X_l \}.
\]
A schematic picture of $s_{X,Y}$ is given in Figure 4b.

Finally, let us extend duality from boundaries to cowordisms.

Let $X$, $Y$ be boundaries, and $\sigma : X \to Y$ be a cowordism. Let us identify $\sigma$ with the underlying multiword $\sigma = (\sigma_0, \sigma_c)$.

The \textit{dual cowordism} $\sigma^\perp : Y^\perp \to X^\perp$ of $\sigma$ is the multiword with the same singular part $\sigma_c$ and the regular part $\sigma^\perp_0$ obtained from $\sigma_0$ by a cyclic permutation of boundary vertices: $\sigma^\perp_0 = \{ [\phi(i), w, \phi(j)] \mid [i, w, j] \in \sigma_0 \}$, where $\phi(i) = i + |Y|$, if $i \leq |X|$, $\phi(i) = i - |X|$, if $i > |X|$.

In a schematic picture, duality is shown in Figure 4c.

The full picture, again, can be recovered by expanding every wire into a parallel cluster. For example, if $X, Y$ are as in (1), the above picture translates to the one in Figure 4d (We defined duality to flip tensor factors precisely in order to have this consistency with “parallel wires substitution” in the graphical language.)

It is very easy to check that, for a fixed alphabet $T$, we have a well-defined category \textbf{Coword}$_T$ of boundaries and cowordisms, and the operation of tensor product together with symmetry cowordisms make it a \textit{symmetric monoidal} category. Moreover there are natural isomorphisms
\[
(X \otimes Y)^\perp \cong X^\perp \otimes Y^\perp \quad \text{Hom}(Y \otimes X, Z) \cong \text{Hom}(X, Y^\perp \otimes Z),
\]
which means that the duality makes the category \textit{compact} (see [KL80], also [AC09]). The first isomorphism in (2) is the symmetry; the second one is shown in Figure 4e.
In fact, in a sense that can be made precise, the category of cowordisms over an alphabet $T$ is a \textit{free compact} category generated by the free monoid $T^*$, where the latter is seen as a category with one object (compare with [Abr05]).

4 Representing linear \(\lambda\)-calculus

Here we assume that the reader is familiar with basic notion of \(\lambda\)-calculus, see [Bar85] for reference.

We assume that we are given sets $X$ and $C$ of \textit{variables} and \textit{constants}, with $C \cap X = \emptyset$. The set $\Lambda = \Lambda(X,C)$ of \(\lambda\)-terms is constructed from $X$ and $C$ by applications and $\lambda$-abstractions.

In linear $\lambda$-calculus, terms are typed using (intuitionistic) \textit{implicational linear logic} (ILL). Given a set $N$ of \textit{literals} or \textit{atomic types}, the set $T_p = T_p(N)$ of \textit{linear implicational types} (over $N$), is defined by the grammar $T_p ::= N | T_p \to T_p$.

A \textit{typing judgement} is a sequent of the form $x_1 : A_1, \ldots, x_n : A_n \vdash t : A$, where $x_1, \ldots, x_n \in X$ are pairwise distinct ($n$ may be zero), $t \in \Lambda(X,C)$, and $A_1, \ldots, A_n, A \in T_p(N)$.

A \textit{linear signature}, or, simply, a \textit{signature}, $\Sigma$ is a triple $\Sigma = (N, C, \Xi)$, where $N$ is a finite set of atomic types, $C$ is a finite set of constants and $\Xi$ is a function assigning to each constant $c \in C$ a linear implicational type $\Xi(c) \in T_p(N)$. We say that $\Sigma$ is a signature \textit{over the set $N$ of atomic types}.

Typing judgements of the form $\vdash c : \Xi(c)$, where $c \in C$, are called \textit{signature axioms} of $\Sigma$.

Typing judgements are derived using type inference rules in Figure 5a (which happen to be \textit{natural deduction} rules of ILL decorated with $\lambda$-terms). Given a signature $\Sigma$, we say that a typing judgement is \textit{derivable in} $\Sigma$ if it is derivable from axioms of $\Sigma$ by rules of linear $\lambda$-calculus. We write in this case $\Gamma \vdash_{\Sigma} t : A$.

It is well known [BBdPH92] that any symmetric monoidal closed category, in particular, a compact closed category, provides a denotational model for linear $\lambda$-calculus (invariant under $\beta\eta$-equivalence). We specialize to the concrete case of the category $\text{Coword}_T$ of cowordisms over the given alphabet $T$. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{typing_rules.png}
\caption{Typing rules (left) and Cowordism representation (right)}
\end{figure}
So, let the sets $N$ and $T$ of literals and terminal symbols respectively be given. An interpretation $\xi$ of linear types over $N$ in $\text{Coword}_{\mathcal{T}}$ consists in assigning to each atomic type $p \in N$ a boundary $\xi(p)$. This is extended to all types in $Tp(N)$ by $\xi(A \rightarrow B) = \xi(A) \downarrow \xi(B)$.

Now, given a linear signature $\Sigma$ over $N$ and $T$, we want to extend the interpretation to derivable typing judgements, so that a judgement of the form $x_1 : A_1, \ldots, x_n : A_n \vdash t : A$ is interpreted as a cowordism of the form $\xi(A_1) \otimes \cdots \otimes \xi(A_n) \rightarrow \xi(A)$.

**Interpretation of typing judgements** consists in assigning, for each constant $c$ and axiom $\vdash c : A$ of $\Sigma$ (here $A = \Xi(c)$), a multiword $\xi(c)$ with boundary $\xi(A)$, which we identify with a cowordism $\xi(c) : 1 \rightarrow \xi(A)$.

This is extended to all typing judgements derivable in $\Sigma$ by induction on type inference rules. The (Id) axiom $x : A \vdash x : A$ is interpreted as the identity cowordism $\text{id}_{\xi(A)}$. Typing judgements obtained by the ($\rightarrow$ I) or ($\rightarrow$ E) rules are interpreted according to Figure 5 (where the symbol $\xi$ is omitted).

In the sequel we often will abuse notation and denote a type in $Tp(N)$ and its interpretation in $\text{Coword}_{\mathcal{T}}$ with the same symbol, as is customary in the literature.

## 5 String abstract categorial grammars

The string signature $\text{Str}_{\mathcal{T}}$ over $T$, where $T$ is a finite alphabet, is the linear signature with a single atomic type $O$, the alphabet $T$ as the set of constants and the typing assignment $\Xi(c) = O \rightarrow O \forall c \in T$. We denote the type $O \rightarrow O$ as $\text{str}$.

Any word $w = a_1 \ldots a_n$ in the alphabet $T$ can be represented as the term $\rho(w) = a_1 \circ \cdots \circ a_n$, where $a_1 \circ \cdots \circ a_n = (\lambda x. a_1(\cdots(a_n(x))\ldots))$, and $\vdash \text{str}_T \rho(w) : \text{str}$.

Moreover, it can be shown that, for any term $t$, if $\vdash \text{str}_T \ t : \text{str}$ then $t \sim_{\beta\eta}^{\text{str}} \rho(w)$ for some $w \in T^*$.

The cowordism representation $\xi_0$ of the string signature $\text{Str}_{\mathcal{T}}$ over the alphabet $T$ is given by the following interpretation in $\text{Coword}_{\mathcal{T}}$.

For the atomic type $O$ we put $\xi_0(O) = \{1\}$, $(\xi_0(O))_1 = \emptyset$. (I.e. $\xi_0(O)$ is a single-point boundary).

Then for each axiom $\vdash c : O \rightarrow O$, where $c \in T$, we put $\xi_0(c) = [1, c, 2]$.

The latter is the multiword with boundary $O^\perp \otimes O$ consisting of a single edge labeled with $c$.

Given linear signatures $\Sigma_i = (N_i, C_i, \Xi_i)$, $i = 1, 2$, a homomorphism of signatures $\phi : \Sigma_1 \rightarrow \Sigma_2$ is a pair of maps

$$\phi_{Tp} : Tp(N_1) \rightarrow Tp(N_2), \ \phi_{Tm} : \Lambda(X, C_1) \rightarrow \Lambda(X, C_2),$$

such that $\phi_{Tm}(A \rightarrow B) = \phi_{Tp}(A) \rightarrow \phi_{Tp}(B)$, $\phi_{Tm}(ts) = (\phi_{Tp}(t)\phi_{Tp}(s))$, $\phi_{Tm}(\lambda x. t) = (\lambda x. \phi_{Tp}(t))$, $\phi_{Tm}(x) = x$ for $x$ a variable, and for any $c \in C_1$ it holds that $\vdash_{\Sigma_2} \phi_{Tm}(c) : \phi_{Tp}(\Xi(c))$. 

9
An abstract categorial grammar over string signature (string ACG) $G$ is a tuple $G = (\Sigma_{\text{abstr}}, T, \phi, S)$, where $\Sigma_{\text{abstr}}$, the abstract signature, is a linear signature, $T$ is a finite alphabet of terminal symbols, $\phi : \Sigma_{\text{abstr}} \to \text{Str}T$, the lexicon, is a homomorphism of signatures, and $S$, the initial type, is an atomic type of $\Sigma_{\text{abstr}}$ with $\phi_T(S) = \text{str}$. We say that $G$ is a string ACG over $T$.

The string language $L(G)$ generated by a string ACG $G$ is the set of words $L(G) = \{ w \in T^* | \exists t \phi_T(t) \sim_{\beta\eta} \rho(w) & c \Sigma_{\text{abstr}}, t : S \}$.

In the setting as above, the cowordism representation $\xi_0$ of $\text{Str}_T$ immediately gives us an interpretation $\xi$ of the abstract signature $\Sigma$ in the category $\text{Coword}_T$ of cowordisms over $T$, obtained as the composition $\xi = \xi_0 \circ \phi$.

That is, for any type $A \in Tp(\Sigma)$ we put $\xi(A) = \xi_0(\phi_T(A))$, and for any signature axiom $\vdash c : \Xi(c)$ of $\Sigma$ we put $\xi(c) = \xi_0(\phi_T(c))$. The latter is a multiword with boundary $\xi(\phi_T(\Xi(c))) = \xi_0(\Xi(c))$.

Because $\phi$ is a homomorphism of signatures, an easy induction on derivation shows that for any typing judgement $x_1 : A_1, \ldots, x_n : A_n \vdash t : A$ derivable in $\Sigma$, its interpretation coincides with the interpretation of the typing judgement $x_1 : \phi_T(A_1), \ldots, x_n : \phi_T(A_n) \vdash \phi_T(t) : \phi_T(A)$ (which is derivable in $\text{Str}_T$).

In particular, for the initial type $S$ we have $\xi(S) = \xi_0(O^1 \otimes O)$ is a two-point boundary, and any derivable typing judgement of the form $\vdash S : t : S$ is interpreted as a single-edge multiword, labeled with $\rho(\phi_T(t))$.

We give a concrete example of a string ACG and its cowordism representation.

We consider the set of atomic types $\{NP, S\}$ and the terminal alphabet $\{\text{John}, \text{Mary}, \text{loves}, \text{madly}, \text{whom}\}$.

The signature axioms and the lexicon are collected in Figures 6a, 6b, while the translation to cowordisms is shown in Figure 6c. We rotated pictures of cowordisms $90^\circ$ counterclockwise, so that outgoing boundaries are shown on the top, with the ordering of vertices from left to right.

We generate the noun phrase “Mary whom John loves madly”, represented as a term of type $NP$. The derivation is shown in Figure 7a for convenience, we break it into five consecutive steps. A step-by-step translation into the language of cowordisms is shown in Figures 7b, 7c with omission of the last step, which should become clear by the end.

6 Linear logic grammars

Recall that, given a set $N$ of positive literals or atoms, the set $Fm = Fm(N)$ of multiplicative linear logic ($\text{MLL}$) formulas over $N$ is defined by the grammar $\text{Lit} := N | N^\perp$, $Fm := \text{Lit} | Fm \otimes Fm | Fm \otimes Fm$.

Connectives $\otimes$ and $\otimes$ are called respectively tensor (also times) and cotensor (also par).

Linear negation $(\cdot)^\perp$ is not a connective, but is definable by induction as $(P^\perp)^\perp = P$, for $P \in N$, and $(A \otimes B)^\perp = B^\perp \otimes A^\perp$, $(A \otimes B)^\perp = B^\perp \otimes A^\perp$.

Note that, somewhat non-traditionally, we follow the convention that negation flips tensor/cotensor factors, typical for noncommutative systems. This
† \textit{JOHN} : NP, † \textit{MARY} : NP, † \textit{LOVES} : NP \to NP \to S, † \textit{MADLY} : (NP \to S) \to NP \to S, † \textit{WHOM} : (NP \to S) \to NP \to NP.

(a) Axioms
\[
\phi_{T_{p}}(NP) = \phi_{T_{p}}(S) = O \to O, \\
\phi_{T_{m}}(\textit{JOHN}) = \text{John}, \quad \phi_{T_{m}}(\textit{MARY}) = \text{Mary}, \quad \phi_{T_{m}}(\textit{JIM}) = \text{Jim}, \\
\phi_{T_{m}}(\textit{LOVES}) = \lambda xy.((y \circ \text{loves} \circ x), \quad \phi_{T_{m}}(\textit{MADLY}) = \lambda fx.((f \cdot x) \circ \text{(madly)}), \\
\phi_{T_{m}}(\textit{WHOM}) = \lambda fx.(x \circ \text{(whom)} \circ (f \cdot (\lambda y.y))).
\]

(b) Lexicon

(c) Cowordism representation

Figure 6: Cowordism representation of a string ACG
Figure 7: ACG representation example
Figure 7: ACG representation example (continued)
\[ \vdash X^\perp, X \quad (\text{Id}), \quad \vdash \Gamma, X \vdash X^\perp \Delta \quad (\text{Cut}), \quad \vdash \Gamma, Y, X, \Delta \quad (\text{Ex}), \quad \vdash \Gamma, Y, X, \Delta \quad (\otimes). \]

(a) MLL sequent calculus

\[ \Gamma, X \quad \vdash \Gamma, Y, X, \Delta \quad (\otimes). \]

(b) Cowordism representation

Figure 8: Cowordism representation of MLL

does not change the logic (the formulas \( A \otimes B \) and \( B \otimes A \) are provably equivalent), but is more consistent with the intended interpretation in the category of cowordisms.

An MLL sequent (over the alphabet \( N \)) is a finite sequence of MLL formulas (over \( N \)). The sequent calculus for MLL [Gir87] is shown in Figure 8a.

It is well known [See89] that semantics of MLL proof theory is provided by \( \ast \)-autonomous categories. Compact categories are a particular (degenerate) case of these, so the category Coword\(_T\) of cowordisms over an alphabet \( T \) allows interpretation of MLL (invariant under cut-elimination).

Just as in the case of linear \( \lambda \)-calculus (and ILL), an interpretation \( \xi \) consists in assigning to every atom \( A \in N \) a boundary \( \xi(A) \). This is extended to all formulas in \( Fm(N) \) by \( \xi(A \otimes B) = \xi(A) \otimes \xi(B) \) and \( \xi(A^\perp) = \xi(A)^\perp \) (note that the extension is well defined).

A sequent \( \Gamma = A_1, \ldots, A_n \) is interpreted as the cotensor of its formulas: \( \xi(\Gamma) = \xi(A_1 \, \bowtie \, \cdots \, \bowtie \, A_n) = \xi(A_1) \otimes \cdots \otimes \xi(A_n) \). A proof \( \sigma \) of the sequent \( \vdash \Gamma \) is interpreted as a multiword with boundary \( \xi(\Gamma) \), which we identify with a cowordism \( \xi(\sigma) : 1 \to \xi(\Gamma) \). Rules for interpreting sequent calculus proofs are represented in Figure 8b (the symbol \( \xi \) omitted and picture rotated counterclockwise with outgoing boundaries on the top, as before).

Given an interpretation \( \xi \) of \( Fm(N) \) in the category Coword\(_T\), we say that a cowordism typing judgement (over \( N \) and \( T \)) is an expression of the form \( \sigma \vdash \Gamma \), where \( \Gamma \) is an MLL sequent (over \( N \)), and \( \sigma : 1 \to \xi(\Gamma) \) is a cowordism (over \( T \)).

A linear logic grammar LLG \( G \) is a tuple \( G = (N, \xi, T, \text{Lex}, S) \), where \( N, \xi, T \) are as above, while \( \text{Lex} \), the lexicon, is a finite set of cowordism typing judgements over \( N \) and \( T \), called axioms, and \( S \in N \); the initial type, is a positive literal with \( |\xi(S)| = 2 \) and \( (\xi(S))_l \) a singleton.

We say that the cowordism typing judgement \( \sigma \vdash \Gamma \) is derivable in \( G \), or that \( G \) generates cowordism \( \sigma \) of type \( \Gamma \) if there exists a derivation of \( \vdash \Gamma \) from axioms of \( G \) whose interpretation is \( \sigma \).
Any regular cowordism of the initial type $S$ generated by $G$ is an edge-labeled graph containing a single edge labeled with a word over $T$. Thus the set of type $S$ regular cowordisms can be identified with a set of words in $T^*$. The language $L(G)$ generated by $G$ is the set of words labeling type $S$ regular cowordisms generated by $G$.

**Theorem 1** A language generated by a string $ACG$ is also generated by an LLG.

**Proof** Given a string $ACG G = (\Sigma, T, \phi, S)$ over the set $N$ of atomic types and the terminal alphabet $T$, we identify types of $\Sigma$ with a subset of the set $F m(N)$ of $MLL$ formulas using the translation $A \triangleright B = A \bot \land B$.

Then the cowordism representation $\xi$ of $G$ gives us an interpretation of $F m(N)$ in $Coword T$. Taking as the lexicon $Lex$ the set of all cowordism typing judgements $\xi(c) \vdash A$ where $c : A$ is an axiom of $\Sigma$, we obtain the LLG $G' = (N, \xi, T, Lex, S)$.

By induction on derivations it can be shown that for any cowordism $\sigma$ of the form $\sigma : \xi(A_1) \otimes \ldots \otimes \xi(A_n) \rightarrow \xi(A)$, where $A_1, \ldots, A_n, A$ are in $T p(N)$, the cowordism typing judgement $\sigma \vdash A_1^., \ldots, A_n^., A$ is derivable in $G'$ iff $\sigma$ is the cowordism representation of some typing judgement $A_1, \ldots, A_n \vdash A$ derivable in $\Sigma$. (Essentially, this repeats the proof that $ILL$ is a conservative fragment of $MLL$.) The statement follows. □

It seems reasonable to ask whether the converse is true. We would expect that the answer is yes, and the formalism of LLG does not add extra expressivity.

### 7 LLG and multiple context-free grammars

We discuss relations between LLG and multiple context-free grammars. Assume that we are given a finite alphabet $N$ of nonzero arity predicate symbols called nonterminal symbols and a finite alphabet $T$ of terminal symbols.

**Production** is a sequent of the form

$$B_1(x_1^{k_1}, \ldots, x_{k_1}, \ldots), \ldots, B_n(x_1^{p_n}, \ldots, x_{p_n}) \vdash A(s_1, \ldots, s_k), (3)$$

where $A, B_1, \ldots, B_n \in N$ have arities $k, k_1, \ldots, k_n$ respectively, $\{x_i^j\}$ are pairwise distinct variables not from $T$, and $s_1, \ldots, s_k$ are words built of terminal symbols and $\{x_i^j\}$, so that each of the variables $x_i^j$ occurs exactly once in exactly one of $s_1, \ldots, s_k$ (here $n$ may be zero).

A multiple context-free grammar (MCFG) $G$ is a tuple $G = (N, T, S, P)$ where $N, T$ are as above, $P$ is a finite set of productions, and $S \in N$, the initial symbol, is unary.

The set of predicate formulas derivable in $G$ is defined by the following induction.
Formula $A(t_1, \ldots, t_k)$ is derivable, if there is a production of the form (3) in $P$, such that $B_1(s^1_1, \ldots, s^1_{k_1}), \ldots, B_n(s^n_1, \ldots, s^n_{k_n})$ are derivable, and $t_m$ is the result of substituting the word $s^j_m$ for every variable $x^j_i$ in $s_m$, $m = 1, \ldots, k$. (The case $n = 0$ is the base of induction.)

The language generated by an MCFG $G$ is the set of words $w$ for which $S(w)$ is derivable. It is well known that any MCFG translates to a string ACG [Sal06], hence to an LLG as well.

A concrete example of a cowordism representation for an MCFG is given in Figure 9. Here we have the terminal alphabet $T = \{a, b\}$, and nonterminal symbols $P, Q$ and $S$ of arities 2, 2 and 1 respectively. The MCFG is defined by the six productions in Figure 9a. It is easy to see that the above generates the language $\{wa^nwb^n | w \in T^*, n \geq 0\}$.

Six cowordisms representing the productions are shown in Figure 9b (for better readability, we label vertices with corresponding variables, the subscripts $l, r$ denoting left and right endpoints respectively). In order to turn these into axioms for an LLG, we have to get rid of the incoming wires. We make all wires outgoing using the bijection $Hom(X, Y) \cong Hom(1, X^\perp \otimes Y)$ (which is a particular case of (2), whose geometric meaning is shown in Figure 4a).

**Theorem 2** A language is generated by an MCFG iff it can be generated by an LLG $G$ with $\otimes$-free lexicon.

**Proof** Translation from MCFG to a $(\otimes$-free) LLG is easy, an example has just been shown. Let us prove the other direction.

For a boundary $X$ and a regular multiword $M$ with boundary $X$, we define the pattern $\text{pat}(M)$ of $M$ as the graph obtained by erasing from $M$ all letters. The set $\text{Patt}(X)$ of all graphs obtained in this way as $M$ varies is the set of possible patterns of $X$. Note that $\text{Patt}(X)$ is finite (maybe empty).

Now, for any $\pi \in \text{Patt}(X)$ choose an enumeration of edges in $\pi$ and introduce a $k$-ary predicate symbol $X^\pi$, where $k$ is the number of edges in $\pi$ (obviously,
\( k \) is the same for all possible patterns of \( X \). Then any regular multiword \( M \) with boundary \( X \) can be unambiguously represented as the predicate formula \( X^\pi(w_1, \ldots, w_k) \), where \( \pi = \text{pat}(M) \), and \( w_i \) is the word labeling the \( i \)-th edge of \( \pi \) in \( M \), \( i = 1, \ldots, k \).

In a similar way, any cowordism \( \sigma : X_1 \otimes \ldots \otimes X_n \rightarrow X \) can be encoded into a finite set of productions. (The above described representation of a multiword is a particular case when \( n = 0 \)).

Fix possible patterns \( \pi_1, \ldots, \pi_n \) of \( X_1, \ldots, X_n \) respectively. There exists at most one possible pattern \( \pi \) of \( X \) such that, whenever \( \text{pat}(M_i) = \pi_i, i = 1, \ldots, n \), it holds that \( \text{pat}(\sigma \circ (M_1 \otimes \ldots \otimes M_n)) = \pi \). If such a \( \pi \) does not exist, then the chosen combination of patterns composed with \( \sigma \) does not produce a regular multiword and is irrelevant for us.

Otherwise choose fresh variables \( x^j_1, j = 1, \ldots, k_j \), where \( k_j \) is the number of edges in \( \pi_i, i = 1, \ldots, n \). Let \( M_i \) be the multiword obtained from \( \pi_i \) by labeling the \( j \)-th edge with \( x^j_1 \). Let \( M = \sigma \circ (M_1 \otimes \ldots \otimes M_n) \). It is a multiword with \( \text{pat}(M) = \pi \). Let \( s_j \) be the word labeling the \( j \)-th edge of \( M \), \( j = 1, \ldots, k \), where \( k \) is the number of edges in \( \pi \). The interaction of \( \sigma \) with the chosen combination of patterns is represented as the production \( X_1^\pi(x^1_1, \ldots, x^1_{k_1}), \ldots, X_n^\pi(x^n_1, \ldots, x^n_{k_n}) \vdash \pi(s_1, \ldots, s_k) \).

Let \( \text{Prod}(\sigma) \) be the set of all productions obtained in this way from \( \sigma \) by varying possible patterns of \( X_1, \ldots, X_n \). Again, note that \( \text{Prod}(\sigma) \) is finite.

Now, let \( G = (N, \xi, T, S, \text{Lex}) \) be a \( \otimes \)-free LLG. The symbol \( \xi \) will be omitted in what follows.

We know that a sequent \( \vdash \Gamma \vdash A \not \vdash B \) is derivable in MLL iff \( \vdash \Gamma, A \vdash B \) is. And since axioms of \( G \) do not use any connective other than \( \not \vdash \), it follows that \( G \) is equivalent to a grammar that does not use any logical connective at all. By cut-elimination, any derivation of the sequent \( \vdash S \) from axioms of \( G \) is equivalent to a derivation not using any logical rule either, i.e. to a one using only the Cut rule.

We construct an equivalent MCFG \( G' = (N', T, S', P) \), by taking the set of nonterminal symbols \( N' = \{ A^\sigma | A \in N \cup N^\perp, \pi \in \text{Patt}(A) \} \), and writing for each axiom \( \alpha \in \text{Lex} \) of the form \( \vdash A_1, \ldots, A_n \), where \( A_1, \ldots, A_n \) are literals, all productions representing cowordisms

\[
\sigma_i : A_{i+1}^\perp \otimes \ldots \otimes A_n^\perp \otimes A_i^\perp \otimes \ldots \otimes A_{i-1}^\perp \rightarrow A_i, \quad i = 1, \ldots, n,
\]

obtained from \( \sigma \) using correspondence \( \text{(2)} \) and symmetry transformations.

We put \( \text{Prod}(\alpha) = \bigcup_i \text{Prod}(\sigma_i) \), and then \( P = \bigcup_{\alpha \in \text{Lex}} \text{Prod}(\alpha) \).

As for the initial symbol \( S' \) of \( G' \), we observe that there is only one possible pattern \( s \) for the boundary \( S \), and we put \( S' = S^s \). An easy induction on derivations shows that \( G \) and \( G' \) generate the same language. □

As a corollary we obtain the known result that any second order ACG generates a multiple context-free language [Sal06]. Thus, we gave a new, geometric proof, arguably quite simple and intuitive.
8 Backpack problem

An LLG of a general form can generate an NP-complete language, just as an ACG (see [YK05]). We give the following, last example as another try to convince the reader that the geometric language of cowordisms is indeed intuitive and convenient for analyzing language generation.

We will consider the backpack problem in the form of the subset sum problem (SSP): given a finite sequence $s$ of integers, determine if there is a subsequence $s' \subseteq s$ such that $\sum_{z \in s'} z = 0$. It is well known [MT90] that SSP is NP-complete.

We will generate by means of an LLG an NP-complete language, essentially representing solutions of SSP.

We represent integers as words in the alphabet \{+,-\}, we call them numerals. An integer $z$ is represented (non-uniquely) as a word for which the difference of + and - occurrences equals $z$. We say that a numeral is irreducible, if it consists only of pluses or only of minuses.

We represent finite sequences of integers as words in the alphabet $T = \{+, -, \cdot\}$, with \cdot interpreted as a separation sign. Thus a word in this alphabet should be read as a list of numerals separated by bullets. When all numerals in the list are irreducible, we say that the list is irreducible.

Now consider two positive literals $H, S$ and interpret each of them as the boundary $X$ of cardinality 2 with $X_l = \{2\}$.

First we construct a system of cowordisms which, together with symmetry transformations, generates all irreducible lists of integers that sum to zero. The three cowordisms $cons : S \otimes S \rightarrow S$, $push : S \otimes S \rightarrow S \otimes S$, $close : 1 \rightarrow S$ are shown in Figure 10a. The cowordism $cons$, by iterated compositions with itself, generates lists with arbitrary many empty slots, then $push$ fill the slots with pluses and minuses (always in pairs), and $close$ closes them. All generated lists will sum to zero, and all irreducible lists summing to zero will be generated.

Now, in order to generate solutions of SSP we need some extra, “deceptive” slots, which contain elements not summing to zero. These slots will be represented by the boundary $H$. The corresponding cowordisms $open_H : H \otimes S \rightarrow S$, $push_+ : H \rightarrow H$, $push_- : H \rightarrow H$, $close_H : 1 \rightarrow H$ are shown in Figure 10b.

Figure 10: Encoding the backpack problem
The cowordism $\text{open}_H$ adds deceptive slots to the list, $\text{push}_- \text{ and push}_+$ fill them with arbitrary numerals, and $\text{close}_H$ closes them.

Although, pedantically speaking, the above system is not an LLG according to our definitions, we obtain an LLG by making all wires outgoing, just as in the above discussion of MCFG. Let us denote the generated language as $L_0$.

It is easy to see that $L_0$ membership problem is, essentially, SSP. More precisely $\text{SSP polynomially reduces to } L_0 \text{ membership problem.}$ Indeed, a sequence $s$ of integers is a solution of SSP iff the corresponding irreducible list is in $L_0$.

It follows that $L_0$ is NP-hard.

On the other hand, it is also easy to see that $L_0$ membership problem is itself in NP. Indeed, in order to show that a list $s$ is in $L_0$ it is sufficient to demonstrate a sequence of cowordisms from Figures 10a, 10b generating $s$, and the number of cowordisms in such a sequence clearly is bounded linearly in the size of $s$. Thus $L_0$ is, in fact, NP-complete.

References

[Abr05] Samson Abramsky. Abstract scalars, loops, and free traced and strongly compact closed categories. In José Luiz Fiadeiro, Neil Harman, Markus Roggenbach, and Jan Rutten, editors, *Algebra and Coalgebra in Computer Science*, pages 1–29, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[AC09] Samson Abramsky and Bob Coecke. Categorical quantum mechanics. In *Handbook of Quantum Logic and Quantum Structures*, pages 261–323. Elsevier, 2009.

[Bar85] Hendrik Pieter Barendregt. *The Lambda Calculus - Its Syntax and Semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1985.

[BBdPH92] P. N. Benton, Gavin M. Bierman, Valeria de Paiva, and Martin Hyland. Linear lambda-calculus and categorial models revisited. In Egon Börger, Gerhard Jäger, Hans Kleine Büning, Simone Martini, and Michael M. Richter, editors, *Computer Science Logic, 6th Workshop, CSL ’92, San Miniato, Italy, September 28 - October 2, 1992, Selected Papers*, volume 702 of *Lecture Notes in Computer Science*, pages 61–84. Springer, 1992.

[BD95] John C. Baez and James Dolan. Higher-dimensional algebra and topological quantum field theory. *Journal of Mathematical Physics*, 36(11):6073–6105, 1995.

[BS11] J. Baez and M. Stay. Physics, topology, logic and computation: a Rosetta Stone. In Bob Coecke, editor, *New Structures for Physics*, pages 95–172, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical foundations for a compositional distributional model of meaning. In *Linguistic Analysis (Lambek Festschrift)*, pages 345–384, 2010.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. The Frobenius anatomy of word meanings I: subject and object relative pronouns. *Journal of Logic and Computation*, 23(6):1293–1317, 2013.

Philippe de Groote. Towards abstract categorial grammars. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 148–155, 2001.

Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

G.M. Kelly and M.L. Laplaza. Coherence for compact closed categories. *Journal of Pure and Applied Algebra*, 19:193–213, 1980.

Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65(3):154–170, 1958.

J. Lambek. Type grammar revisited. In Alain Lecomte, François Lamarche, and Guy Perrier, editors, *Logical Aspects of Computational Linguistics*, pages 1–27, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

Vedrana Mihaliček and Carl Pollard. Distinguishing phenogrammar from tectogrammar simplifies the analysis of interrogatives. In Philippe de Groote and Mark-Jan Nederhof, editors, *Formal Grammar*, pages 130–145, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., USA, 1990.

R.A. Muskens. Separating syntax and combinatorics in categorial grammar. *Research on Language and Computation*, 5(3):267–285, 2007. Pagination: 17.

Sylvain Salvati. Encoding second order string ACG with deterministic tree walking transducers. In Shuly Wintner, editor, *The 11th conference on Formal Grammar*, FG Online Proceedings, pages 143–156, Malaga, Spain, 2006. Paola Monachesi; Gerald Penn; Giorgio Satta; Shuly Wintner, CSLI Publications.

R.A.G. Seely. Linear logic, *-autonomous categories and cofree coalgebras. In *Categories in Computer Science and Logic*, pages 371–382. American Mathematical Society, 1989.
[SMFK91]  Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.

[Sto16]  Robert E. Stong. *Notes on Cobordism Theory*. Princeton University Press, 2016.

[YK05]  Ryo Yoshinaka and Makoto Kanazawa. The complexity and generative capacity of lexicalized abstract categorial grammars. In Philippe Blache, Edward Stabler, Joan Busquets, and Richard Moot, editors, *Logical Aspects of Computational Linguistics*, pages 330–346, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.