Bell States and Negative Sentences in the Distributed Model of Meaning

Anne Preller [1]

Informatique
LIRMM/CNRS
Montpellier, France

Mehrnoosh Sadrzadeh[2]

Computing Laboratory
Oxford University
Oxford, United Kingdom

Abstract

We use Bell states to provide compositional distributed meaning for negative sentences of English. The lexical meaning of each word of the sentence is a context vector obtained within the distributed model of meaning. The meaning of the sentence lives within the tensor space of the vector spaces of the words. Mathematically speaking, the meaning of a sentence is the image of a quantizing functor from the compact closed category that models the grammatical structure of the sentence (using Lambek Pregroups) to the compact closed category of finite dimensional vector spaces where the lexical meaning of the words are modeled. The meaning is computed via composing eta and epsilon maps that create Bell states and do substitution and as such allow the information to flow among the words within the sentence.

Keywords: Compact Closed Categories, Pregroups, Vector Spaces, Distributed Model of Meaning, Linguistics, Bell States.

1 Introduction

Why present a paper that belongs to computational and mathematical linguistics in a workshop on quantum physics? Surprisingly, there are intuitive and technical similarities. Maybe insight can be gained by comparing the two approaches.

Protocols for human communication have two aspects, namely transforming the stored information to words (semantics) and fitting the words into a sentence (syntax). Both aspects are performed by the speaker Alice who detains the information and the listener Bob who wants to receive it, but not in the same order. Alice puts the meaning into words according to the rules of the syntax. Bob recognizes the string of words as a sentence,

[1] Email: preller@lirmm.fr

[2] Email: mehrs@comlab.ox.ac.uk

[3] Support by EPSRC (grant EP/F042728/1) and LIRMM is gratefully acknowledged, as are helpful discussions with Claudia Casadio, Stephen Clark, and Bob Coecke. Special thanks goes to Louise Crane who gave us the idea of a quantizing functor.

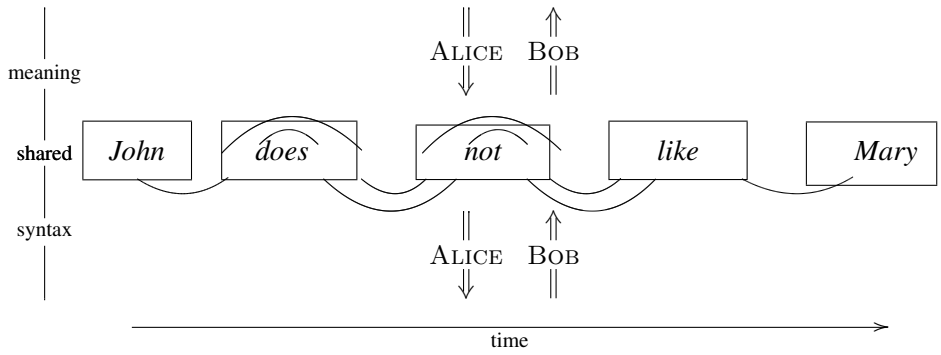


Fig. 1. ALICE INFORMS BOB

reconstructs the meaning of the words and fits them together to form the meaning of the sentence. The shared states of the process are the words, they carry both meaning and syntax. The semantic content of the words is stored in the memory. Meanings represent the 'prepared' bits of the process. The syntactic structure is recognized during processing. The recognition represents the 'observed' bits of the process. This intuitive similarity is underlined by a common mathematical axiomatisation of both the communication and quantum logic protocols. The communication protocols we present use pregroup grammars for syntax and compact closed categories for semantics.

Mathematical Linguists study the mathematical structure of natural languages in terms of their syntax and semantics. Some of the very same mathematical structures have also been used in Computer Science and Physics. For example, Lambek's *Syntax Calculus* [8] is a residuated monoid, later expanded with lattice operations and turned into a 'Quantale'. The term 'Quantale' was introduced by Mulvey as a quantum version (i.e. non-commutative) of the notion of a Locale [11]. They were used to axiomatize an Operational Quantum Logic [6]. In Computer Science, Quantales are algebraic models of Linear Logic [17] and have also been used in logics for concurrency [2].

Recently, some Theoretical Physicists and Mathematical Linguists have independently abandoned the monoidal structure of Quantales for the more expressive setting of compact closed categories. Lambek has used the setting of a *compact bi-category* [14], referred to as a *Pregroup* [10]; these have been applied to analyze syntax of many natural languages, from English and French to Japanese, Arabic, Persian and many others. Abramsky and Coecke [1] have used compact closed categories to provide semantics for quantum protocols and as such have set a new basis for Quantum Logic. Similarities between models of Language and Physics have been pointed out by Lambek in [9].

Apart from syntax, these similarities also occur in the semantic models of natural languages, ranging from logical to distributed models of meaning. From the logical point of view, a category-theoretical semantics for pregroup grammars have been proposed in [13] in the form of compact bi-categories. From the distributed point of view, vector spaces are used to provide lexical meaning for words [16]. Moreover, the Quantum axiomatic of Hilbert spaces have been used to model semantics of natural languages in [18,19]. These models have found applications in information retrieval from documents, for example those on the web, and to find synonymous meanings for words [7].

The logical models of meaning are compositional: the meaning of a sentence is a func-

tion of the meanings of its parts, but these models do not say much about the meanings of the individual words. On the contrary to these, the distributed models of meaning provide a nice semantics for the individual words, but are not compositional. Developing a compositional distributed model of meaning is one of the open problems of the field of semantics of natural languages. Following a proposal by S. Clark and Pulman [5], namely that the vector space tensor product is a promising candidate to to compose meaning vectors, S. Clark, Coecke and the second author provided a solution to this problem in the context of compact closed categories [4]. The mathematical setting was the product category of a *Pregroup* and the category of finite dimensional vector spaces, so the objects were pairs of a linguistic type from the pregroup part and its meaning as a vector within a context vector space. The pairwise tensor of this category was used to compose the meaning of words in a sentence. This method was tested on some simple positive sentences, where the epsilon maps were used to substitute the meaning vectors of the subject and object into the arguments of the linear map modeling a verb. Providing meaning for more complex sentences where logical connectives such as "not" and "and" were involved were left for future work. In this paper, we build on previous work as follows

- We tidy up the mathematical structure of previous work: instead of working in the product category, we work with the more elegant and more natural notion of a "quantizing functor": the functor from the *lexical pregroup dictionary* of a language seen as a free compact bi-category, as constructed in [14], to the compact closed category of finite dimensional vector spaces $FVect$, as used to model Quantum protocols in [1].
- Inspired by the work of the first author in [13] and later in [15], we show how the meaning of the logical connective "not" can be formalized by using *index* types in pregroups and eta maps in $FVect$, these are the co-units of the adjunction on the objects and create Bell states. In this context, they use the freedom provided by the indexes to create extra argument space for linear maps of "does" and "not". This process allows the information to flow from the subject, which is at the beginning of the sentence, to the verb, which as a result of negation is being moved further away from the subject. This is similar to what happens in the *teleportation-based* Quantum protocols such as entanglement swapping. The graphical calculus depicts this flow in a pleasingly simple and clear way and turns the complicated calculations of matrixes into the enjoyable task of pulling ropes or combing hair!
- We take the first step towards developing a logic for semantic derivations in natural languages. Motivated by the work of D. Clark in [3], we develop notation for a graded implication and use it to measure the degree of similarity between positive and negative sentences. Meanings of sentences can be derived from one another using this implication and the degree of this implication stands for how close the meanings of the sentences are to each other.

2 Background

2.1 Compact Closed Categories

A compact closed category is a monoidal closed category with the product $\otimes$ and its unit I, whenever for each object A there are also objects A^r and A^l, and morphisms

$$\eta^l : \mathrm{I} \to A \otimes A^l \qquad \epsilon^l : A^l \otimes A \to \mathrm{I}$$

$$\eta^r : \mathrm{I} \to A^r \otimes A \qquad \epsilon^r : A \otimes A^r \to \mathrm{I}$$

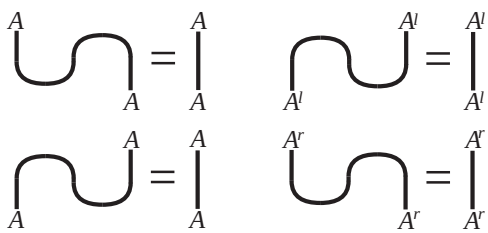
which satisfy:

$$(1_A \otimes \epsilon^l) \circ (\eta^l \otimes 1_A) = 1_A \quad (\epsilon^l \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l) = 1_{A^l}$$

$$(\epsilon^r \otimes 1_A) \circ (1_A \otimes \eta^r) = 1_A \quad (1_{A^r} \otimes \epsilon^r) \circ (\eta^r \otimes 1_{A^r}) = 1_{A^r}$$

When depicting the morphisms $\eta^l, \epsilon^l, \eta^r, \epsilon^r$ as

these axioms simplify to



i.e. they boil down to 'yanking wires' or 'combing hair'. The free compact closed category $T(\mathcal{B})$ generated by a partially ordered set $\mathcal{B}$ exists, this free construction has been spelled out in [14]. If a compact closed category is symmetric then we have the extra symmetry natural isomorphisms $\sigma_{A,A'} : A \otimes A' \to A' \otimes A$. In this category the left and right adjoints become identity.

2.2 Pregroup Grammars

Let Σ be the set of words of a natural language and $\mathcal{B}$ a partially ordered set. A *Pregroup dictionary* for Σ based on $\mathcal{B}$ is a binary relation $D \subseteq \Sigma \times T(\mathcal{B})$, where $T(\mathcal{B})$ is the free compact 2-category generated over the partial order $\mathcal{B}$. We refer the reader for the details of this construction to the joint work of the first author with J. Lambek in [14]. Every element (w, t) of dictionary D is called a *lexical entry* in D.

A *Pregroup grammar* $G = \langle D, s \rangle$ for Σ based on $\mathcal{B}$ consists of a dictionary D and a distinguished elements $s \in \mathcal{B}$. A string of words $w_1 \ldots w_n$ of Σ is said to be *grammatical* if and only if $f : t_1 \cdots t_n \to s$ is a morphism of $T(\mathcal{B})$, where each (w_i, t_i) is a lexical entry in D. These morphisms are sometimes referred to as *reductions*.

For example and as suggested in [15], we consider a pregroup grammar for English with the following entries in its pregroup dictionary; it generates sentences "John likes Mary" and "John does not like Mary".

John	:	n	does	:	$n^r \otimes s \otimes j^l \otimes \sigma$
likes	:	$n^r \otimes s \otimes n^l$	not	:	$\sigma^r \otimes j \otimes j^l \otimes \sigma$
Mary	:	n	like	:	$\sigma^r \otimes j \otimes n^l$

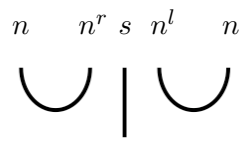
The basic types n, s, j stand for noun phrase, statement and infinitive; σ plays a role similar to an index 'sort' in HPS grammars of [12]. The set $\mathcal{B} = \{n, s, j, \delta\}$ is ordered by equality. Based on these types, the above sentences are grammatical; their reductions are morphisms in $T(\mathcal{B})$. The reduction morphism of "John likes Mary" is

$$\epsilon_n^r \otimes id_s \otimes \epsilon_n^l$$

and has the following type

$$n \otimes (n^r \otimes s \otimes n^l) \otimes n \to s$$

It is depicted as follows in the diagrammatic language of compact closed categories



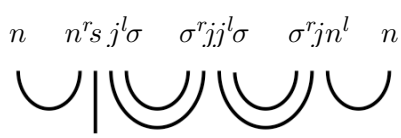
The reduction morphism of "John does not like Mary" is

$$\epsilon_j^l \otimes \epsilon_j^l \circ \epsilon_n^r \otimes id_{sj^l} \otimes \epsilon_\sigma^r \otimes id_{jj^l} \otimes \epsilon_\sigma^r \otimes id_j \otimes \epsilon_n^l$$

and has the following type

$$n \otimes (n^r \otimes s \otimes j^l \otimes \sigma) \otimes (\sigma^r \otimes j \otimes j^l \otimes \sigma) \otimes (\sigma^r \otimes j \otimes n^l) \otimes n \to s$$

It is depicted as follows



2.3 Distributional Model of Meaning

In the distributed model of meaning, the lexical meaning of words are vectors in a possibly high dimensional vector space; one whose bases are certain words of a dictionary. Given a text or a collections of texts and fixing a neighborhood window of n words, one counts how many times a certain word appears in that window in the context of the bases. This provides us with a vector, that is the vector of the lexical meaning of that word.

As an example [4], consider the word *dog* and a vector space with bases *eat*, *sleep*, *pet*, and *furry*. If the word *dog* has *eat* in its context 6 times (in some text), *sleep* 5 times, *pet* 17 times, and *furry* 8 times, then the vector for *dog* in this space is (6,5,17,8). The advantage of representing meanings in this way is that the vector space gives us a notion of distance

between words, so that the inner product (or some other measure) can be used to determine how close in meaning one word is to another. For example, one can form the vector of *cat* in the same space as that of *dog* and then observe that they have similar meanings in that context, which makes sense since cats and dogs are both pets and they both sleep, run and are furry.

Computational models along these lines have been built using large vector spaces (tens of thousands of context words/basis vectors) and large bodies of text (up to a billion words in some experiments). Experiments in constructing thesauri using these methods have been relatively successful. For example, the top 10 most similar nouns to *introduction*, according to the system of [7], are *launch, implementation, advent, addition, adoption, arrival, absence, inclusion, creation*.

2.4 Finite Dimensional Vector Spaces

Consider the category **FVect** of finite dimensional vector spaces and linear maps: objects V are finite dimensional vector spaces over the base field $\mathbb{R}$, morphisms are linear maps, monoidal tensor is the vector space tensor whose unit is the base field of the vector space, and the adjoint of each vector space V is its dual or conjugate space V^*. Since the vector space models of meaning have fixed basis, we assume that each vector spaces comes with an inner product. For a vector space V with base $\{e_i\}_i$ we set $V^l = V^r = V^*$ and obtain that $FVect$ is a compact closed category. The unit and counit of adjunction are as follows

$$\eta^l = \eta^r : \mathbb{R} \to V \otimes V :: 1 \mapsto \sum_i e_i \otimes e_i$$

and

$$\epsilon^l = \epsilon^r : V \otimes V \to \mathbb{R} :: \sum_{ij} c_{ij}\, \psi_i \otimes \phi_j \mapsto \sum_{ij} c_{ij} \langle \psi_i | \phi_j \rangle \,.$$

The epsilon maps are the inner-product extended by linearity to the whole tensor product and eta maps produce Bell states.

3 A Semantic Functor to Quantize Language

For a pregroup dictionary $D \subseteq \Sigma \times T(\mathcal{B})$ and a finite dimensional vector space $FVect$, let the following

$$[\![\,]\!] : T(D) \to FVect$$

be a strongly monoidal functor that moreover satisfies $[\![t^l]\!] = [\![t]\!]^* = [\![t^r]\!]$ for t an object of $T(D)$. We refer to $T(D)$ as the *free dictionary category*, the objects of it are lexical entries, i.e. they constitute of a pair of a natural language word and its grammatical type in the language of pregroups. In TQFT terms, the functor $[\![\,]\!]$ *quantizes* the free dictionary category of the language. We refer to it s our *semantic functor*.

For instance, based on the above pregroup types, this functor may assign the following vector spaces to their corresponding lexical entries:

(John,n)	:	V	(does,$n^r \otimes s \otimes j^l \otimes \sigma$)	:	$V^* \otimes S \otimes J^* \otimes V$
(likes, $n^r \otimes s \otimes n^l$)	:	$V^* \otimes S \otimes W^*$	(not, $\sigma^r \otimes j \otimes j^l \otimes \sigma$)	:	$V^* \otimes J \otimes J^* \otimes V$
(Mary, n)	:	W	(like, $\sigma^r \otimes j \otimes n^l$)	:	$V^* \otimes J \otimes W^*$

Intuitively speaking, one may think of the verb "likes" as the map $V \times W \to S$ that inputs two arguments of the type V and W respectively and outputs a vector from the vector space S. By the universal property of tensor, to each such map corresponds a linear map $V \otimes W \to S$. Since $FVect$ is closed, to this linear map corresponds a vector (i.e. the name of the linear map) in $V \otimes S^* \otimes W$, isomorphic to its dual space $V^* \otimes S \otimes W^*$. Similarly, "like" can be seen as the map that inputs two vectors from spaces V and W but produces an infinitive of the type J. The auxiliary verb "does" creates identical correlations: it inputs an infinitive verb and returns the same infinitive. "not" creates opposite correlations by inputing an infinitive and outputting it with the same values in opposite bases.

Meaning of a positive transitive sentence

As shown in previous work [4], given the above meaning spaces for each word, the meaning vector and vector space of the sentences "John likes Mary" is simply obtained by calculating the map of its syntactic reduction in the semantic category $FVect$. The semantic version of this map is as follows

$$\epsilon_V \otimes 1_S \otimes \epsilon_W : V \otimes (V^* \otimes S \otimes W^*) \otimes W \to S$$

Its diagram is the same as the diagram for the syntactic reduction of the sentence, that is

The meaning vector of this sentence is a vector in S. For $\overrightarrow{John} \in V, \overrightarrow{Mary} \in W, \overrightarrow{likes} \in V^* \otimes S \otimes W^*$, it is calculated as follows

$$\overrightarrow{John\ \ likes\ \ Mary} \quad = \quad \left(\langle \epsilon_V | \otimes 1_S \otimes \langle \epsilon_W |\right) \left|\overrightarrow{John} \otimes \overrightarrow{likes} \otimes \overrightarrow{Mary}\right\rangle$$

Given that $\overrightarrow{likes}$ lives in a tensor space, it can be written as follows

$$\overrightarrow{likes} = \sum_{ikj} C_{ikj}\, \overrightarrow{v}_i \otimes \overrightarrow{s}_k \otimes \overrightarrow{w}_j \in V \otimes S \otimes W$$

So the above Dirac expression is equal to the following

$$\sum_{ikj} C_{ikj} \langle \overrightarrow{John} | \overrightarrow{v}_i \rangle \overrightarrow{s}_k \langle \overrightarrow{w}_j | \overrightarrow{Mary} \rangle \quad = \quad \sum_k \left(\sum_{ij} C_{ik} \langle \overrightarrow{John} | \overrightarrow{v}_i \rangle \langle \overrightarrow{w}_j | \overrightarrow{Mary} \rangle \right) \overrightarrow{s}_k \,.$$

One may get more concrete by assuming that the vector space V is spanned by all men $\{\overrightarrow{m_i}\}_i$ and the vector space W by all women $\{\overrightarrow{f_j}\}_j$. A Boolean truth-value meaning to

the above sentence is obtained by assuming that S is spanned by two vectors $|1\rangle$ and $|0\rangle$, denoting *true* and *false*. The verb "likes" becomes the following superposition

$$\overrightarrow{likes} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{s}_{ij} \otimes \overrightarrow{f_j}$$

where $\overrightarrow{s}_{ij} = |1\rangle$ if m_i likes f_j and $\overrightarrow{s}_{ij} = |0\rangle$ otherwise. Assuming that John is m_3 and Mary is f_4, the meaning of our sentence becomes

$$\sum_{ij} \langle \overrightarrow{m}_3 \mid \overrightarrow{m_i} \rangle \otimes \overrightarrow{s}_{ij} \otimes \left\langle \overrightarrow{f}_j \mid \overrightarrow{f_4} \right\rangle = \sum_{ij} \delta_{3i} \overrightarrow{s}_{ij} \delta_{j4} = \overrightarrow{s}_{34}$$

and is *true* if John likes Mary and *false* otherwise. The epsilon maps act like substitution: they substitute the values for vectors "John" and "Mary", that is $\overrightarrow{m}_3$ and $\overrightarrow{f_4}$, in their place holders in the vector of "likes" $\overrightarrow{m_i}$ and $\overrightarrow{f_i}$.

Meaning of a negative transitive sentence

Computing the meaning vector and vector space of the negative version of the above sentence is more involved. This is because the auxiliary verb "does" and the negation preposition "not" come between the verb "like" and its subject "John". As a result, there will be a distance between the verb and its subject and the substitutions that computed the meaning of the positive version of the sentence cannot go through anymore. One solution to this problem has been proposed and used by the first author in providing semantics for pregrousp in [15]. In the setting of vector spaces, the map of the syntactic reduction is pre-composed with eta maps and linear maps of meaning of logical words (in this case "does" and "not") to allow the information flow among the non-adjacent words within the sentence and be logically acted upon. The eta maps produce the spaces of the index types of the lexical entries of the words and let the computation to proceed via substitutions. In the language of QM [1], the eta maps create Bell states that produce extra space and allows for *teleportation*, that is they enable the information to flow between the quantum states that are not locally close.

The process of computing meaning has thus two steps: we first apply some eta maps and then compute the syntactic reduction map of the sentence. The map of the first step has the following types

$$g\colon V\otimes(V^*\otimes J\otimes W^*)\otimes W \to V\otimes V^*\otimes(S\otimes J^*)\otimes V\otimes V^*\otimes(J\otimes J^*)\otimes V\otimes(V^*\otimes J\otimes W^*)\otimes W$$

and is given below, $\otimes$'s are dropped, $\overline{does}$ and $\overline{not}$ are meanings of "does" and "not"

$$(1_{VV^*S}\overline{does}1_{VV^*J}\overline{not}1_{VV^*JW^*W}) \circ (1_{VV^*}\eta_{S=J}1_{VV^*}\eta_J 1_V 1_{V^*JW^*}1_W) \circ (1_V\eta_V\eta_V 1_{V^*JW^*}1_W)$$

The first composite of the above creates Bell states $\eta_V \otimes \eta_V$ for teleporting "John" into "likes"; the second composite creates Bell states $\eta_{S=J}$ for the base swapping vector of "not" and η_J for the identity vector of "does". The result is then composed with the map of the syntactic reduction of the negative sentence, depicted on page 3, as follows

$$f\colon V \otimes (V^* \otimes S \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes W^*) \otimes W \to S$$

and is given by

$$(1_S \otimes \epsilon_J \otimes \epsilon_J) \circ (\epsilon_V \otimes 1_S \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes \epsilon_W)$$

The full map of the meaning is obtained by the composition of the above two steps as the map $f \circ g$, which has the following types

$$V \otimes (V^* \otimes J \otimes W^*) \otimes W \to V \otimes V^* \otimes (S \otimes J^*) \otimes V \otimes V^* \otimes (J \otimes J^*) \otimes V \otimes (V^* \otimes J \otimes W^*) \otimes W \to S$$

Concretely, the meaning of the sentence "John does not like Mary" is calculated as follows: we assume that both of the vector spaces J and S are spanned by the same two vectors as before and thus are the same vector spaces, i.e. $S = J$. Vector spaces V to which "John" belongs and W to which "Mary" belongs are spanned as in the positive case above. Since "like" in the negative sentence is the infinitive form of the verb and thus cannot produce a sentence, its vector changes to

$$\overrightarrow{like} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{\mu}_{ij} \otimes \overrightarrow{f_j} \ \in \ V^* \otimes J \otimes W^* \qquad \text{where} \qquad \overrightarrow{\mu}_{ij} = \begin{cases} |1\rangle & m_i \text{ likes } f_j \\ |0\rangle & o.w. \end{cases}$$

The vector of "not" uses the Bell state $|10\rangle + |01\rangle$, which swaps the bases

$$\overrightarrow{not} \ = \ \sum_k \overrightarrow{m}_k \otimes (|10\rangle + |01\rangle) \otimes \overrightarrow{m}_k \ \in \ V^* \otimes J \otimes J^* \otimes V$$

The vector of "does" uses the Bell state $|11\rangle + |00\rangle$, which acts as identity on the bases

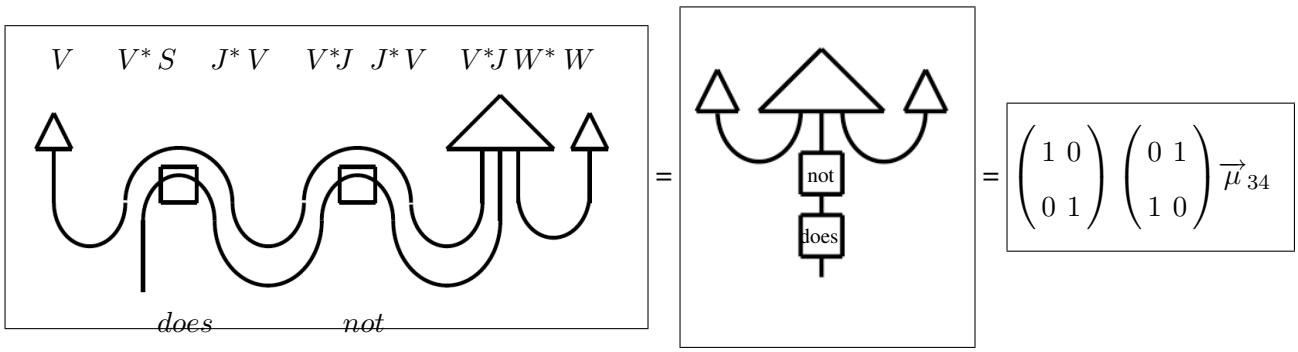
$$\overrightarrow{does} \ = \ \sum_l \overrightarrow{m}_l \otimes (|11\rangle + |00\rangle) \otimes \overrightarrow{m}_l \ \in \ V^* \otimes S \otimes J^* \otimes V$$

Assuming that John is m_3 and Mary is f_4 and abbreviating $|10\rangle + |01\rangle$ to $\overline{not}$ and $|00\rangle + |11\rangle$ to $\overline{does}$, the meaning of the sentence is calculated by applying the meaning map $(f \circ g)$ to the tensor product of the meanings of the words within the sentence, that is

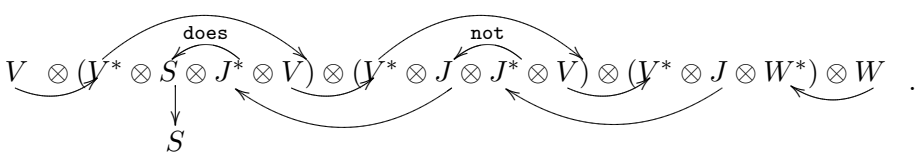
$$\begin{aligned}
&(f \circ g)\Big(\overrightarrow{m_3} \otimes \big(\textstyle\sum_l \overrightarrow{m}_l \otimes \overline{does} \otimes \overrightarrow{m}_l\big) \otimes \big(\textstyle\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \overrightarrow{f}_j\big) \otimes \overrightarrow{f}_4\Big) &= \\
&\big(\textstyle\sum_l \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_l\rangle \otimes \overline{does} \otimes \overrightarrow{m}_l\big) \otimes \big(\textstyle\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \langle \overrightarrow{f}_j \mid \overrightarrow{f}_4\rangle\big) &= \\
&\big(\textstyle\sum_l \delta_{3l} \otimes \overline{does} \otimes \overrightarrow{m}_l\big) \otimes \big(\textstyle\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \delta_{j4}\big) &= \\
&\overline{does} \otimes \overrightarrow{m}_3 \otimes \big(\textstyle\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\big) &= \\
&\overline{does} \otimes \big(\textstyle\sum_k \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_k\rangle \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\big) &= \\
&\overline{does} \otimes \big(\textstyle\sum_k \delta_{3k} \otimes \overline{not} \otimes \overrightarrow{m}_k\big) \otimes \big(\textstyle\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\big) &= \\
&\overline{does} \otimes \overline{not} \otimes \overrightarrow{m}_3 \otimes \big(\textstyle\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\big) = \overline{does} \otimes \overline{not} \otimes \big(\textstyle\sum_i \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_i\rangle \otimes \overrightarrow{\mu}_{i4}\big) &= \\
&\overline{does} \otimes \overline{not} \otimes \big(\textstyle\sum_i \delta_{3i} \otimes \overrightarrow{\mu}_{i4}\big) = \overline{does} \otimes \overline{not} \otimes \overrightarrow{\mu}_{34} = (|00\rangle + |11\rangle) \otimes (|10\rangle + |01\rangle) \otimes \overrightarrow{\mu}_{34} &= \\
&|0010\overrightarrow{\mu}_{34}\rangle + |0001\overrightarrow{\mu}_{34}\rangle + |1110\overrightarrow{\mu}_{34}\rangle + |1101\overrightarrow{\mu}_{34}\rangle = |01\overrightarrow{\mu}_{34}\rangle + |10\overrightarrow{\mu}_{34}\rangle &= \\
&\begin{cases} |011\rangle + |101\rangle & \overrightarrow{\mu}_{34} = 1 \\ |010\rangle + |100\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases} = \begin{cases} |0\rangle & \overrightarrow{\mu}_{34} = 1 \\ |1\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases}
\end{aligned}$$

That is, the meaning of "John does not like Mary" is true whenever $\overrightarrow{\mu}_{34}$ is false, that is whenever the meaning of "John likes Mary" is false. Intuitively, we are first computing the value of the linear map corresponding to "likes", that is $V \otimes W \to S$ by substituting in its arguments the values for John and Mary. Then we compute the value of the linear map corresponding to "not" that is $J \to J$, by substituting in its argument the value computed by "likes". Finally, we substitute this value into the argument of the linear map corresponding to "does", that is $J \to S$.

The above calculation is depicted as follows in the diagrammatic language of [1], where the triangles are the states of a quantum system involved in an informatique protocol:



The first diagram above gets a more informative shape in the diagrammatic 2-categorical language of [13,14], where the arrows are oriented and thus the flow of information is depicted more clearly:



The swinging curls of this diagram are the same as those of figure 1. The top swinging line of eta's teleports "John" into "likes", the bottom swinging line of eta's applies the "not" and "does" vectors. The top swinging line of epsilons shows the domino-like substitutions of "John" into "does", "not", and "like", the bottom swinging line does so for "does", "not", and "like".

4 Comparing Meaning of Sentences

In the previous section, we assigned a truth-value meaning to our sentences. One can also consider degrees of truth or falsity. For example, in previous work [4], we assumed 'like' has degrees of "love" and "hate" by making S to be spanned by two vectors $\overrightarrow{l}$ and $\overrightarrow{h}$, defined as follows

$$\overrightarrow{loves} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{loves}_{ij} \otimes \overrightarrow{f_j}\,, \qquad \overrightarrow{hates} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{hates}_{ij} \otimes \overrightarrow{f_j}$$

where now $\overrightarrow{loves}_{ij} = \overrightarrow{l}$ if m_i loves f_j and $\overrightarrow{loves}_{ij} = \overrightarrow{0}$ otherwise, and $\overrightarrow{hates}_{ij} = \overrightarrow{h}$ if m_i hates f_j and $\overrightarrow{hates}_{ij} = \overrightarrow{0}$ otherwise. Now we may define the verb "likes" to have degrees of "love" and "hate", for instance as follows

$$\overrightarrow{likes} = \frac{3}{4}\overrightarrow{loves} + \frac{1}{4}\overrightarrow{hates}$$

So the meaning of "John likes Mary" for m_3 and f_4 as "John" and "Mary" becomes the vector $\begin{pmatrix} 3/4 & 1/4 \end{pmatrix}$ in the vector space whose basis are "love" and "hate". These degrees propagate to the negative case and the meaning of "John does not like Mary" is obtainable by applying the Bell state of not to μ_{34}, that is

$$(|01\rangle + |10\rangle) \begin{pmatrix} 3/4 & 1/4 \end{pmatrix} = \begin{pmatrix} 1/4 & 3/4 \end{pmatrix}$$

One of the advantages of our approach to compositional meaning is that the meaning of sentences are all vectors in the same space, so one can use the inner product to compute their degree of similarity. In previous work, we used this tool to compare meaning of sentences of the same type, that is, positive transitive sentences. In particular, we compared meaning of sentences such as "John likes Mary" to "John loves Mary" and "John hates Mary". Here we show how the meaning of negative sentences can be compared to other negative sentences, also to other positive sentences. For example, we compare the meaning of "John does not like Mary" to "John does not love Mary", but also to "John likes Mary" and "John loves Mary".

Given the lexical entries for the two grammatical string of words $\alpha = t_1, t_2, \ldots, t_n$ and $\beta = t'_1, t'_2, \cdots, t'_m$, we say they are p close iff $\langle [\![\alpha]\!] \mid [\![\beta]\!] \rangle = p$. Here are some examples from previous work for comparing the meaning of different positive transitive sentences

$$\langle [\![\overrightarrow{m}_3 \otimes \overrightarrow{loves} \otimes \overrightarrow{f}_4]\!] \mid [\![\overrightarrow{m}_3 \otimes \overrightarrow{likes} \otimes \overrightarrow{f}_4]\!] \rangle = \frac{3}{4}$$

$$\langle [\![\overrightarrow{m}_3 \otimes \overrightarrow{loves} \otimes \overrightarrow{f}_4]\!] \mid [\![\overrightarrow{m}_3 \otimes \overrightarrow{hates} \otimes \overrightarrow{f}_4]\!] \rangle = 0$$

Now we are in the position to also compare the meaning of positive and negative sentences, here are some examples

$$\langle [\![\overrightarrow{m}_3 \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{f}_4]\!] \mid [\![\overrightarrow{m}_3 \otimes \overrightarrow{loves} \otimes \overrightarrow{f}_4]\!] \rangle = \frac{1}{4}$$

$$\langle [\![\overrightarrow{m}_3 \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{f}_4]\!] \mid [\![\overrightarrow{m}_3 \otimes \overrightarrow{hates} \otimes \overrightarrow{f}_4]\!] \rangle = \frac{3}{4}$$

$$\langle [\![\overrightarrow{m}_3 \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{f}_4]\!] \mid [\![\overrightarrow{m}_3 \otimes \overrightarrow{likes} \otimes \overrightarrow{f}_4]\!] \rangle = \frac{3}{8}$$

And so on.

References

[1] S. Abramsky, B. Coecke, 'A categorical semantics of quantum protocols', *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Science Press, 2004.

[2] S. Abramsky, S. Vickers, 'Quantales, observational logic and process semantics', *Mathematical Structures in Computer Science* **3**, 1993.

[3] D. Clark, *Context-Theoretic Semantics for Natural Language*, Ph.D. Thesis, University of Sussex, September 2007.

[4] S. Clark, B. Coecke, M. Sadrzadeh, 'A Distributional Compositional Model of Meaning', *Proceedings of Conference on Quantum Interactions*, P. Bruza, W. Lawless, J. van Rijsbergen (eds.), College Publications, University of Oxford, March 2008.

[5] S. Clark, S. Pulman, 'Combining Symbolic and Distributional Models of Meaning', *Proceedings of AAAI Spring Symposium on Quantum Interaction*, AAAI Press, 2007.

[6] B. Coecke, D. J. Moore, I. Stubbe', 'Quantaloids describing causation and propagation of physical properties', *Foundations of Physics Letters* **14**, 2001.

[7] J. Curran, *From Distributional to Semantic Similarity*, University of Edinburgh, 2004.

[8] J. Lambek, 'The mathematics of sentence structure', *American Mathematics Monthly* **65**, 1958.

[9] J. Lambek, 'Compact Bi-categories in Linguistics and Physics", to appear in the *Proceedings of Cats Kets Cliosters*, Oxford, 2006.

[10] J. Lambek, *From Word to Sentence*, Polimetrica, Milan, July 2008.

[11] C. J. Mulvey, &, *Supplemento ai Rendiconti del Circolo Matematico di Palermo***II**, 1992.

[12] C. Pollard, I. Sag, *Head-driven phrase structure grammar*, University of Chicago Press, 1994.

[13] Preller Anne, *Category Theoretical Semantics for Pregroup Grammars*, Philippe Blache and Edward Stabler (Eds.): Logical Aspects of Computational Linguistics **3492**(2005), pp. 254-270.

[14] A. Preller, J. Lambek 'Free compact 2-categories', *Mathematical Structures in Computer Science* **17**, 2007.

[15] A. Preller, 'Towards Discourse Representation via Pregroup Grammars', *Journal of Logic Language Information* **16**, January 2007.

[16] H. Schuetze, 'Automatic Word Sense Discrimination', *Computational Linguistics* **24**, 1998.

[17] D.N. Yetter, 'Quantales and (non-commutative) Linear Logic', *Journal of Symbolic Logic* **55**, 1990.

[18] C. J. van Rijsbergen, *The Geometry of Information Retrieval*, Cambridge University Press 2004.

[19] D. Widdows, *Geometry and Meaning*, Center for the Study of Language and Information/SRI, 2004.