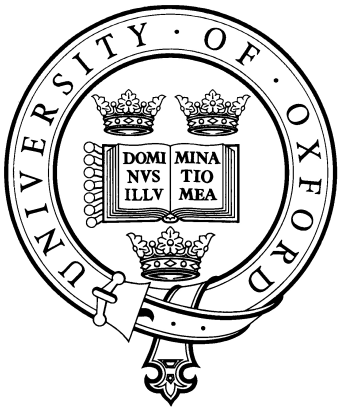
Encoding Logical Words as Quantum Gates: The Higher Dimensional Case



Marco Fernandez

marco.fernandez@stx.ox.ac.uk; fernandez@maths.ox.ac.uk

St. Cross College

University of Oxford

A thesis submitted in partial fulfilment of the requirements for the degree of

MSc in Mathematics and Foundations of Computer Science

September 2010

Supervisors: Bob Coecke, Mehrnoosh Sadrzadeh

Oxford University Computing Laboratory

bob.coecke@comlab.ox.ac.uk, mehrnoosh.sadrzadeh@comlab.ox.ac.uk

Contents

Acknowledgements

I want to express my sincere gratitude to both supervisors Dr. Bob Coecke and Dr. Mehrnoosh Sadrzadeh from the Oxford University Computing Laboratory (OUCL) with their support and patience for the regular meetings. On several occasions they lasted more than two hours and sometimes also lead to very long abstract emails from my side and helpful response. It is their work, expanded from work in recent years by Professor Pulman, head of the computational linguistics group, that provides the basis for this thesis and the very strong proposal of a mathematical framework that unifies the compositional theory of grammatical types with a distributional theory of meaning. I highly appreciate that the opportunity was granted to pursue something new and on my own with many parts that may never have been done before.

During the research for the thesis I also got to consult the advice of Professor Pulman who enlightened me with his expertise in the field of computational linguistics, theoretical linguistics and logics. I was also granted the chance to give an additional short presentation of the initial work in the computational linguistics seminar where Professor Pulman gave me a very good insight into a logical and philosophical issue of meaning. It was him and Dr. Stephen Clark, at Oxford University's Computing Laboratory before and then at Cambridge's, who came up with the first ideas for such a framework combining symbolic and distributional models of meaning [7]. This had a major influence on the main work pursued and for all the support I am very grateful.

I want to thank Edward Grefenstette who is pursuing his DPhil at the Computing Laboratory towards the same mathematical and computational goal, however using Combinatory Categorial Grammar (CCG) and by setting up the mathematical meaning by the numbers of grammatical relations to the other constituents of a sentence. His thorough explanations and data base provide the logical background for some of the types associated to the basic sentences under consideration.

Furthermore, I also want to thank Prof. Claudia Casadio, Prof. Joachim Lambek, Dr. Dominic Widdows, Prof. Samson Abramsky and Prof. Laurence R. Horn for the brilliant contributions to the main papers and books that provided the input for the new result in the thesis.

0.1 Motivation and Origins

An important feature in quantum computer science and computational linguistics is the involvement of quantum gates, the analogue to the classical logic gates to build up digital circuits. They play a vital role in many quantum protocols, some of which are significantly more efficient in comparison to classical algorithms, e.g. in search algorithms and they also provide the blocks needed for constructions of quantum computers. Moreover, in the analysis and implementation of computational linguistics in categorial grammars they can similarly be implemented and, for instance, be used to calculate or compare the meaning of sentences. This has been introduced in an algebraic approach from a computational linguistics and logics point of view in [4] and [11]. The mathematical framework for the quantum theoretical application is expanded in [6].

Joachim Lambek introduced the algebraic structure of **pregroups** for analysing the syntax of individual languages, see [11]. Together with **vector spaces** they can be used as a mathematical framework to associate and calculate meanings of sentences in various philologies and even to compare the meanings of different sentences. The paper by both the supervisors [6] applies this to determine the two-dimensional truth-theoretic meaning by implementing "**NOT**" in analogy to the Pauli X-gate in quantum computer science:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \rightarrow \qquad X \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} b \\ a \end{pmatrix}$$

This is then also expressed in the diagrammatic calculus as introduced in the quantum computer science course. It corresponds to a basis or coefficient swapping for the quantum superposition and is, therefore, essentially two-dimensional and cannot be generalised uniformly in the same manner. The development of a canonical matrix of negation that works uniformly for higher dimensions is left open as future work and is pursued in this master thesis. The situation is similar for other logical words, such as "And" or "If then".

An approach in information retrieval for uniform expansion to higher dimensions is suggested in [17], by means of projections to orthogonal subspaces.

Abstract

The open problem and topic to be covered in this master thesis is the development of logical words with focus on "NOT" as quantum gates in the three and higher dimensional cases. This is undertaken in the associated framework of vector spaces, pregroups and the diagrammatic calculus of compact closed categories as in [6]. It involves the development of canonical matrices or other maps that work uniformly for higher dimensions. The maps are then validated by means of implementation in well-chosen grammatical sentences in several languages as well as their mathematical meaning calculations.

Various approaches will be inspected using linguistic and quantum mechanical intuition. One idea that is investigated was first expressed by supervisor Mehrnoosh Sadrzadeh by having a multi-dimensional gate in the form of a linear map; e.g. to be able to distinguish mathematically the semantic difference among the various uses of these logical words, such as predicative or non-predicative use of the associated verb. This leads to both syntactic, but also semantic differences, such as having or not having a noun phrase as the header of their linguistic tree diagram (cf. [9]).

For negation the distinction does not accurately apply as worked out in the thesis, hence the main philosophical and logical interpretations of negation will be investigated. Higher-dimensional maps are set up for each one of them. In some cases it will be proven that no such *linear* maps exist and non-linear maps are set up instead. A second investigation consists in another possible way for an expansion to higher dimension as aforementioned by means of projections to orthogonal subspaces. This is then also implemented in the diagrammatic calculus.

Future work can then contain expansion to other important logical words and sentences in the same manner and application in the categorical approach as undertaken by supervisor Bob Coecke and several of his DPhil students. Eventually, this can contribute significantly to quantum computer science, computational linguistics, information retrieval and A.I. by implementation in quantum protocols and machine translation.

Chapter 1

Research Procedure

I attended the excellent quantum computer science course by Dr. Andreas Doering and Dr. Bob Coecke during Hilary term with substantial reading in the topic and a significant amount of time invested into the final mini-project and current research. Over the spring break the potential supervisors were contacted and current research papers on the topic (viz. [4], [6], [15] and [17]) identified, read and discussed with the two supervisors to determine the open problem that was to be pursued. The questions took on their current shape in further discussion and analysis during the first two weeks of Trinity term together with the two supervisors as well as advice for the computational linguistics aspect of the thesis from Professor Pulman and DPhil student Edward Grefenstette at the OUCL. Trinity term was invested to pursue the research via the approaches explained in regular contact with the supervisors, besides English linguistics preparation for the GRE and one further successful course at the Mathematics Department. This led to less time available than for co-students in the course, but to successfully getting the last required exam score for a possible distinction depending on a good thesis of 80%. Writing up was partly undertaken on the way for the drafts and results. The final evaluation and major writing up was then done from mid-July throughout August with the viva voce outstanding on 24th September, 2010.

Table 1.1 shows the plan for completion of the research.

Timeline	Work	Progress
	Contact supervisors and define the topic in proposal	completed
May.	Reading and research	completed
July/August	Thesis writing	completed
September	Viva voce	outstanding

Table 1.1: Plan for completion of my research

Chapter 2

Introduction
Encoding Logical Words as Quantum Gates: The Higher Dimensional Case

2.1 Syntax and Semantics

Definition 2.1.1 (Natural Language)**.** *A* natural language *is a language that has developed naturally in use, as contrasted with an artificial language or computer code.*[1]

The scientific study of such natural languages is undertaken by the field of linguistics and computational implementations thereof in the field of computational linguistics. There are two important sub-fields of linguistics of interest in this thesis, namely the study of the structure of the language (grammar) and the study of meaning. They can also be analysed in mathematical linguistics as syntax and semantics, respectively.

[1]Def. as in: ”natural language”, Oxford Dictionaries, *Oxford University Press*, April 2010.

Syntax:

A classic way to analyse the syntax of a language is the Chomsky[2] way by formulating so called "rewrite rules" for the grammatical rules of a language.

An alternative, modern way is by an algebraic approach using type-logical (categorial) grammars. The latter approach proved mathematically very efficient and, hence will be applied in this framework. Here types are associated to the constituents of a phrase and these are then composed in a second step in order to get the type of the whole phrase.

Semantics:

To analyse the semantics of natural languages there are two very distinct ways of approach.

1.) One can either go by **symbolic meaning**; whereby sets are assigned to the types of the language and these are then connected using logical connectives.

2.) On the other hand, one can associate high-dimensional vector spaces to the constituents of sentences and thereby derive a so called **distributional meaning** approach. The bases are selected appropriately to the domain of meaning.

Both methods have their advantages and disadvantages. The first one is compositional, but only qualitative in the sense of yielding true-false. It gives rise to model-theoretic semantics as developed by the logician Richard Montague. This is based largely on higher-order predicate calculus and the lambda calculus, allowing the cross to formal languages. However, it is not so effective for lexical semantics or for real world texts.

The second one, on the other hand, is the opposite by being quantitative, but not compositional. For the sub-field of lexical semantics it is efficient.

[2]Noam Chomsky is one of the main contributors to linguistics of the 20th century and professor emeritus of linguistics at MIT.

A modern, sophisticated way for an overall analysis of a natural language was suggested by the mathematician Joachim Lambek with his **syntactic calculus** that is highly regarded in the logic and linguistic community (see [6] and [11]). In the last few years he refined his syntactic calculus to the algebraic structure of so called **pregroups** that allow to analyse the structure of natural languages in an algebraic setting. It has been successfully applied to many languages, such as English, French, Arabic, Italian and Persian (by supervisor Sadrzadeh in [15]) over the last few years. This is particularly interesting because pregroups share a common structure with vector spaces and tensor products in category theory, as will be shown in the last part of this thesis. In their very recent paper from this year ([6], 2010) the supervisors achieve to unite the benefits of the two opposing approaches outlined above by relying on types from pregroups to model the syntax and on vector spaces to model the semantics so that using the tensor product to combine the two structures even the meaning of whole sentences can be calculated. The application of the tensor product is analogue to the use in Hilbert spaces in quantum computer science.

The categorial unification thereof gives a very strong unified framework for the field. In addition, implementation in the **diagrammatic calculus** allows a significant simplification of the calculation and higher efficiency.

Each step of the procedure for the meaning association to sentences will be discussed in detail in the following sections. A short overview is as follows:

In a first step, a type p_i is associated to each word w_i of a sentence to make up for the **syntax**. These types can then be reduced by using the axioms and rules of the associated pregroup grammar.

The second step then consists in also assigning a vector space to each constituent of a sentence depending on its type assignments to cover the **semantics** of the sentences.

In a final step, the **tensor product** is applied to the created vector spaces for all the words in the sentences. In combination with the associated **diagram of the syntactic reduction**, this allows to efficiently evaluate the meaning of sentences.

2.2 Pregroups

This gives the algebraic setting of pregroups to model the syntax as set up and defined by Lambex in [11].

Definition 2.2.1 (Semigroup)**.** *A* semigroup *is an algebraic structure consisting of a set $S \neq \emptyset$ together with a binary operation, denoted by juxtaposition, that satisfies the associative law:*

$$(xy)z = x(yz).$$

Definition 2.2.2 (Monoid)**.** *A* monoid *is a semigroup with an identity element 1 such that:*

$$x1 = x = 1x.$$

Definition 2.2.3 (Partially Ordered)**.** *A set is* partially ordered*, if it is equipped with a binary operation $\leq$ such that:*

$$x \leq x \text{ (reflexive law)},$$
$$\text{if } x \leq y \text{ and } y \leq z, \text{ then } x \leq z \text{ (transitive law)},$$
$$\text{if } x \leq y \text{ and } y \leq x, \text{ then } x = y \text{ (antisymmetry law)}.$$

In the case of a semigroup or monoid to be partially ordered one also requires the following:

$$\text{if } x \leq y \text{ and } x' \leq y', \text{ then } xx' \leq yy' \text{ (compatibility)},$$

or equivalently :

$$\text{if } x \leq y, \text{ then } uxv \leq uyv \text{ (substitutivity)}.$$

Definition 2.2.4 (Left Adjoint)**.** *An element x in a partially ordered monoid (S, $\leq$, $\cdot$, 1) is said to have a* left adjoint *denoted x^l, if:*

$$x^l x \leq 1 \leq x x^l.$$

Lemma 2.2.5. *Left adjoints are unique.*

Proof. As usual for uniqueness proofs proceed by starting with another left adjoint as follows. Let x^m be such another left adjoint of x, then:

$$x^m = x^m 1 \leq x^m(xx^l) = (x^m x)x^l \leq 1x^l = x^l.$$

Analogously, $x^l \leq x^m$.
Then it follows from the antisymmetry law that: Since $x^m \leq x^l$ and $x^l \leq x^m$, then $x^m = x^l$. □

Definition 2.2.6 (Left Pregroup)**.** *A partially ordered monoid in which every element has a left adjoint is called a* left pregroup*.*

Theorem 2.2.7. *For a partially ordered monoid:*

$$\textit{If } x \leq y, \textit{ then } y^l \leq x^l \textit{ (contravariance).}$$

Proof. Assume that $x \leq y$, then using substitutivity, one gets the following:

$$y^l = y^l 1 \leq y^l x x^l \leq y^l y x^l \leq 1 x^l = x^l.$$

□

Furthermore, double contravariance yields: If $x \leq y$, then $x^{ll} \leq y^{ll}$.

Definition 2.2.8 (Group)**.** *A* group *is a monoid with every element x having a left inverse x^{-1} such that $x^{-1}x = 1$.*

Theorem 2.2.9. *For a group the left inverse is also a right inverse.*

Proof. Consider the left inverse x^{-1} for x so that $x^{-1}x = 1$, then also:

$$xx^{-1} = 1xx^{-1} = (x^{-1})^{-1}x^{-1}xx^{-1} = (x^{-1})^{-1}1x^{-1} = (x^{-1})^{-1}x^{-1} = 1.$$

□

Remark: If one were to use groups for the mathematical analysis, then e.g. assigning a type, such as $n_i {n_i}^{-1} = 1$ (for $i = 1, 2, 3$) to one of the constituents of a sentence would allow to place that constituent anywhere in the sentence and still reduce the sentence to s making it grammatical, if it was mathematically grammatical as in definition 2.3.3 without it. The reduction when assigning a type from a pregroup such as $n_i {n_i}^l$ instead, will depend on the neighbouring constituents. That is one of the main reasons why so called *pregroups* are used in the linguistic application. All this is defined and derived in the following.

Definition 2.2.10 (Right Pregroup)**.** *A* right pregroup *is a partially ordered monoid in which each element x has a* right adjoint x^r *with:*

$$xx^r \leq 1 \leq x^r x.$$

Definition 2.2.11 (Pregroup)**.** *A* pregroup *is both a left and a right pregroup.*

Theorem 2.2.12. *In a pregroup every adjoint is unique and order reversing, the latter meaning contravariance as follows:*

$$\textit{If } x \leq y, \textit{ then } y^l \leq x^l \textit{ and } y^r \leq x^r$$

Proof. This follows from lemma 2.2.5, theorem 2.2.7 and the analogue proofs for right adjoints. □

Definition 2.2.13. *Furthermore, denote the left adjoint of x^r as: x^{rl}, and analogously for the opposite.*

Theorem 2.2.14. *For a pregroup it also holds that opposite adjoints annihilate each other:*

$$x^{rl} = x = x^{lr}.$$

Proof. Since $xx^r \leq 1 \leq x^r x$, then x is some left adjoint of x^r. Moreover, since left adjoints are unique, it follows that x is the left adjoint of x^r. □

Example 2.2.15. *Examples of pregroups are on the one hand obviously* partially ordered groups*, meaning partially ordered monoids where each element has an inverse which serve as the adjoints.*

Proof. A group is a monoid in which every element has a left inverse, therefore the left pregroup property is satisfied. By theorem 2.2.9 it follows that every left inverse is also a right inverse, and hence the right pregroup property is also satisfied. □

Remark: A pregroup that is not a partially ordered group is, for instance, the monoid of all order preserving maps from $\mathbb{Z}$ to $\mathbb{Z}$.

Definition 2.2.16 (Free Pregroup)**.** *As defined in [2]: "Let $(P, \leq)$ be a partially ordered set of primitive types, $P^{(\mathbb{Z})} = \{p^{(i)} | p \in P, i \in \mathbb{Z}\}$ is the set of atomic types and* $\text{Cat}_{(P,\leq)} = (P^{(\mathbb{Z})})^* = \{p_1^{(i_1)} \cdots p_n^{(i_n)} | 1 \leq k \leq n, p_k \in P, i_k \in \mathbb{Z}\}$ *is the set of [compound] types. For $X, Y \in$* $\text{Cat}_{(P,\leq)}$*, $X \leq Y$ iff this is deducible in the system for pregroup grammars as below where: $p, q \in P; n, k \in \mathbb{Z}$, and $X, Y, Z \in$* $\text{Cat}_{(P,\leq)}$*. Wojciech Buszkowski [3] has proposed this construction to define a pregroup that extends $\leq$ on P to* $\text{Cat}_{(P,\leq)}$*.*

$$X \leq X(Id),$$

$$\frac{XY \leq Z}{Xp^{(n)}p^{(n+1)}Y \leq Z}(A_L),$$

$$\frac{Xp^{(k)}Y \leq Z}{Xq^{(k)}Y \leq Z}(IND_L),$$

$$\frac{X \leq Y \quad Y \leq Z}{X \leq Z}(Cut),$$

$$\frac{X \leq YZ}{X \leq Yp^{(n+1)}p^{(n)}Z}(A_R),$$

$$\frac{X \leq Yq^{(k)}Z}{X \leq Yp^{(k)}Z}(IND_R).$$

Whereby for the last one: $q \leq p$, if k is even and $p \leq q$, if k is odd."

Remark: In the categorial setting as well as in [11] the binary operator $\rightarrow$ is used for a partial order, instead of the mathematically reserved $\leq$.

2.3 Pregroup Grammars

Let $\mathcal{B}$ be a partially ordered set and Σ be the set of words of a natural language. Also let $T(\mathcal{B})$ be the *free pregroup* that is generated over the partial ordered set $\mathcal{B}$ as constructed by Lambek.

Definition 2.3.1 (Pregroup Dictionary)**.** *A* pregroup dictionary *for Σ on the underlying set $\mathcal{B}$ is a binary relation:*

$$D \subseteq \Sigma \times T(\mathcal{B}).$$

Definition 2.3.2 (Pregroup Grammar)**.** *A* pregroup grammar *is a pair of a pregroup dictionary and a set of distinguished elements $\alpha \subset \mathcal{B}$:*

$$G = \langle \mathrm{D}, \alpha \rangle.$$

Remark: It was also Buszkowski who, using the so called *Switching Lemma*, proved that pregroup grammars are context-free, see [11].

Definition 2.3.3 (Grammatical Sentence)**.** *Let $w_1 \dots w_n$ be a string of words in Σ with each $(w_i, t_i) \in \mathrm{D}$. Such a string is* grammatical *if and only if*

$$t_1 \cdots t_n \leq s \in \alpha, \qquad \textit{for } t_i \in T(\mathcal{B}),\ \forall i = 1, \dots, n.$$

Example 2.3.4. *Let us consider some of the main examples of grammatical sentences discussed in this thesis: "Bob likes Diet-Coke" and the negation thereof: "Bob does not like Diet-Coke". One can set up a dictionary that generates these sentences using the primitive types* n, j, s, a, o *and* σ*, abbreviations for noun, infinitive, statement, attribute, object and index type, respectively. They then build up further atomic types as follows* $\mathrm{n}^l, \mathrm{n}^r, \mathrm{j}^l, \mathrm{j}^r$ *... and compound types, such as* $\mathrm{n}^r\mathrm{s}\mathrm{n}^l$*. As set above a sentence will be grammatical according to the mathematical definition, if the juxtaposition of the associated types reduces to a statement* s *(or a question).*

The following types are associated to the constituents of the phrase so that the sentences are grammatical. The reductions are morphisms in $T(B)$.

$$\begin{aligned} Bob &: n \\ likes &: n^r s n^l \\ Diet\text{-}Coke &: n \\ does &: n^r s j^l \sigma \\ not &: \sigma^r j j^l \sigma \\ like &: \sigma^r j n^l \end{aligned}$$

Proof. We can show that based on these types the above sentences are in fact grammatical by considering the following composition for the sentences in discussion and their reductions. Note that the brackets are only for simplification to the reader. The linear maps of meaning in the graphical calculus are drawn besides them.

$$\begin{array}{cccccc} Bob & likes & Diet\text{-}Coke. & \to & statement \\ n & (n^r s n^l) & n & \leq & s \end{array}$$

$$\begin{array}{cccccccc} Bob & does & not & like & Diet\text{-}Coke. & \to & statement \\ n & (n^r s j^l \sigma) & (\sigma^r j j^l \sigma) & (\sigma^r j n^l) & n & \leq & s \end{array}$$

□

Chapter 3

Logical Negation

An approach for modelling the logical NOT-gate was worked out in a categorical approach to distributed meaning by the two supervisors of this thesis Dr. Bob Coecke and Dr. Mehrnoosh Sadrzadeh at the Oxford University Computing Laboratory and Dr. Stephen Clark then at the University of Cambridge Computer Laboratory. This also contains a last investigation of the distinction between predicative and non-predicative position of the NOT in a grammatical sentence for its meaning. As an example, the following shows the distinction that is made:

Bob does not like Diet-Coke $\cong$ *Bob dislikes Diet-Coke.*

It is not the case that Bob likes Diet-Coke $\ncong$ *Bob dislikes Diet-Coke.*

The second because the meaning expressed can be that Bob in this case may be indifferent to Diet-Coke or may prefer another beverage more.

3.1 Meaning Evaluation

Definition 3.1.1. *Define the vector of the* $\overrightarrow{w_1 \cdots w_n}$ *of the meaning of a string of words* $w_1 \cdots w_n$ *to be the following:*

$$\overrightarrow{w_1 \cdots w_n} := f(\overrightarrow{w_1} \otimes \cdots \otimes \overrightarrow{w_n}).$$

The linear map f *is constructed by substituting each type* $p_i \in [p_1 \cdots p_n \leq x]$ *with the associated meaning* W_i *in* (W_i, p_i)*, the meaning space of the word.*

Remark: If the string of words forms a grammatical sentence, then the reduced type x of the sentence should be the basic grammatical type s of a sentence, as defined beforehand.

The linear maps that were worked out by the supervisors are essentially 1, 2 or 3-dimensional and in the last case having free parameters. The following will discuss these and set up the required definitions and examples for the expansion to higher dimension afterwards.

One can span the vector space V by all men $\{\overrightarrow{m}_i\}_i$ and W by all kinds of (inorganic) matter $\{\overrightarrow{d}_j\}_j$. Then for the two-dimensional or Boolean meaning case applied one spans the truth-space S by two vectors denoted in Dirac notation as $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ for true and false, respectively. This results in the following superposition for the verb $\overrightarrow{likes}$:

$$\overrightarrow{likes} = \sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{s}_{ij} \otimes \overrightarrow{d}_j.$$

Note: The following consideration is for the set of grammatical sentences with the negations associated to a verb. This does not include ungrammatical sentences as used for instance in English sometimes, e.g.: "Not so much", which is short for some grammatical sentence, such as "I do not want so much".

Definition 3.1.2. *In order to get a truth-theoretic meaning define:*

$$\overrightarrow{s}_{ij} = \begin{cases} |1\rangle, & \text{if } m_i \text{ likes } d_j. \\ |0\rangle, & \text{otherwise.} \end{cases}$$

Example 3.1.3. *As an example consider one of the fundamental questions discussed in quantum physics. Let the famous physicists:* $\overrightarrow{Christiaan\ Huygens}$, $\overrightarrow{Issac\ Newton}$ *and* $\overrightarrow{Thomas\ Young}$ *be elements in* V*. These correspond to the* $m_i \in V$ *in a chronological order. Similarly,* W *may contain:* $\overrightarrow{Particles}$, $\overrightarrow{Waves}$ *and* $\overrightarrow{Duality}$ *as* d_j*, for* $i, j = 1, 2, 3$ *in that order. Then the meaning of sentences expressing which physicist likes or came up with which hypothesis for the composition and propagation of light can be encoded using the Kronecker function as follows:*

$$\sum_{ij} \langle \overrightarrow{m}_1 | \overrightarrow{m}_i \rangle \otimes \overrightarrow{s}_{ij} \otimes \langle \overrightarrow{d}_j | \overrightarrow{d}_2 \rangle = \sum_{ij} \delta_{1i} \overrightarrow{s}_{ij} \delta_{j2} = \overrightarrow{s}_{12} \quad = |1\rangle.$$

$$\sum_{ij} \langle \overrightarrow{m}_2 | \overrightarrow{m}_i \rangle \otimes \overrightarrow{s}_{ij} \otimes \langle \overrightarrow{d}_j | \overrightarrow{d}_1 \rangle = \sum_{ij} \delta_{2i} \overrightarrow{s}_{ij} \delta_{j1} = \overrightarrow{s}_{21} \quad = |1\rangle.$$

$$\sum_{ij} \langle \overrightarrow{m}_3 | \overrightarrow{m}_i \rangle \otimes \overrightarrow{s}_{ij} \otimes \langle \overrightarrow{d}_j | \overrightarrow{d}_3 \rangle = \sum_{ij} \delta_{3i} \overrightarrow{s}_{ij} \delta_{j3} = \overrightarrow{s}_{33} \quad = |1\rangle.$$

3.2 Negative Transitive Sentence

Investigate the main negated transitive sentence under consideration:

$$\overrightarrow{Bob}\ \overrightarrow{does}\ \overrightarrow{not}\ \overrightarrow{like}\ \overrightarrow{Diet\text{-}Coke}.$$

Following [6]: "The meaning space of the auxiliary verb is $(V \otimes S \otimes J \otimes V, n^r s j^l \sigma)$, that of the negation particle is $(V \otimes J \otimes J \otimes V, \sigma^r j j^l \sigma)$, and that of the verb is $(V \otimes J \otimes W, \sigma^r j n^l)$. The 'from-meaning-of-words-to-meaning-of-a-sentence' linear map f as developed in [6] in this case is:

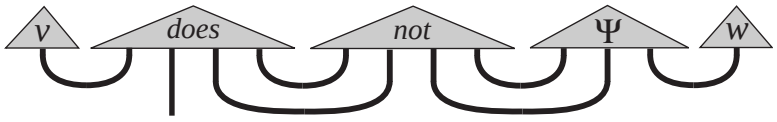
$$f = (1_S \otimes \epsilon_J \otimes \epsilon_J) \circ (\epsilon_V \otimes 1_S \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes \epsilon_W) :$$
$$V \otimes (V^* \otimes S \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes W^*) \otimes W \to S$$

This is drawn in the graphical calculus as follows:

When applied to the meaning vectors of words one obtains:

$$f(\overrightarrow{v} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{\Psi} \otimes \overrightarrow{w}),$$

which is depicted as:



The trick to implement *NOT* will be to take this linear map to be the linear matrix representing the logical *NOT*. Concretely, while the matrix of the identity is $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, the matrix of the logical *NOT* is $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. In Dirac notation, the vector corresponding to the identity is $|00\rangle + |11\rangle$, while the vector corresponding to the logical *NOT* is $|01\rangle + |10\rangle$. Where for $\overrightarrow{does}$ set $S = J$, then:

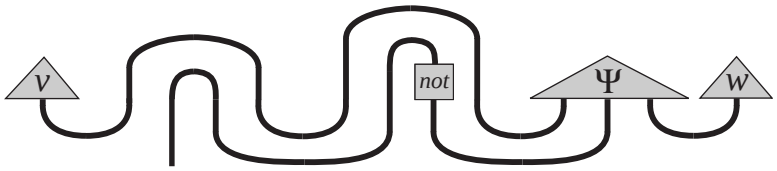
$$\overrightarrow{does} \ = \ \sum_i \overrightarrow{e}_i \otimes (|00\rangle + |11\rangle) \otimes \overrightarrow{e}_i \ \in \ V \otimes J \otimes J \otimes V \, ,$$

We will set:

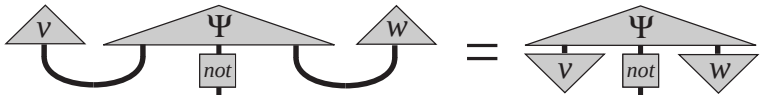
$$\overrightarrow{not} \;=\; \sum_i \overrightarrow{e}_i \otimes (|01\rangle + |10\rangle) \otimes \overrightarrow{e}_i \;\in\; V \otimes J \otimes J \otimes V \,.$$

Diagrammatically we have:

Substituting all of this in $f(\overrightarrow{v} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{\Psi} \otimes \overrightarrow{w})$ one obtains, diagrammatically:

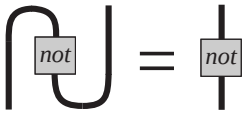


By the diagrammatic calculus of compact closed categories this is equal to:



(3.1)

This is a consequence of the fact that from the axioms for categories one has:



The configuration on the left always encodes the transpose and the matrix of the *NOT* is obviously self-transpose. The two "cups" then cancel out as a direct consequence of the axioms. In the language of vectors and linear maps, the left-hand side of eq. (3.1) is:

$$\left(\epsilon_V \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \epsilon_W\right) (\overrightarrow{v} \otimes \overrightarrow{\Psi} \otimes \overrightarrow{w}) \,.$$

Remark: The above pictures are very similar to the ones encountered in quantum information protocols such as quantum teleportation and entanglement swapping (see [13] and the extensive papers by supervisor Bob Coecke). There the morphisms η and ϵ encode Bell-states and corresponding measurement projectors.

3.3 Negative Truth-Theoretic Meaning.

The meaning of the sentence

$$\overrightarrow{Bob\ does\ not\ like\ Diet\text{-}Coke.}$$

is then calculated as follows. Assume that the truth-space S is spanned by the vectors $|0\rangle$ and $|1\rangle$, as in the previous example. Set $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\overrightarrow{Bob} = \overrightarrow{m}_1 \in V$, $\overrightarrow{Diet\text{-}Coke} = \overrightarrow{d}_4 \in W$.

Furthermore, the vector of *like* is also equivalent to before:

$$\overrightarrow{like} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{like}_{ij} \otimes \overrightarrow{d_j} \qquad \text{for} \qquad \overrightarrow{like}_{ij} = \begin{cases} |1\rangle, & \text{if } m_i \text{ likes } d_j. \\ |0\rangle, & \text{otherwise.} \end{cases}$$

Then apply the *from meaning of words to meaning of a sentence* linear map f. One gets the simple diagrammatic reduction shown before, and thereby obtains the following straightforward evaluation:

$$(\epsilon_V \otimes X \otimes \epsilon_W)\left(\overrightarrow{m}_1 \otimes \overrightarrow{likes} \otimes \overrightarrow{d}_4\right) = \sum_{ij} \langle \overrightarrow{m}_1 \mid \overrightarrow{m}_i \rangle\, X(\overrightarrow{likes}_{ij}) \langle \overrightarrow{d}_j \mid \overrightarrow{d}_4 \rangle =$$

$$\sum_{ij} \delta_{1i}\, X(\overrightarrow{likes}_{ij})\, \delta_{j4} \quad = \quad X(\overrightarrow{likes}_{14}) =$$

$$\begin{cases} |1\rangle, & \text{if } \overrightarrow{like}_{14} = |0\rangle. \\ |0\rangle, & \text{if } \overrightarrow{like}_{14} = |1\rangle. \end{cases} \quad = \quad \begin{cases} |1\rangle, & \text{if } m_1 \text{ does not like } d_4. \\ |0\rangle, & \text{otherwise.} \end{cases}$$

One can also go through the whole calculation in Dirac notation as follows. Abbreviate $|10\rangle + |01\rangle$ to $\overrightarrow{not}$ and $|00\rangle + |11\rangle$ to $\overrightarrow{does}$, and set $f = h \circ g$ with

$$h = 1_J \otimes \epsilon_J \otimes \epsilon_J \quad \text{and} \quad g = \epsilon_V \otimes 1_J \otimes 1_J \otimes \epsilon_V \otimes 1_J \otimes 1_J \otimes \epsilon_V \otimes 1_J \otimes \epsilon_W$$

The meaning of the sentence is then calculated on the next page by applying the meaning map f to the tensor product of the meanings of the words within the sentence.

$$
\begin{aligned}
& f\left(\overrightarrow{m_1}\otimes(\sum_l \overrightarrow{m}_l\otimes\overrightarrow{does}\otimes\overrightarrow{m}_l)\otimes(\sum_k \overrightarrow{m}_k\otimes\overrightarrow{not}\otimes\overrightarrow{m}_k)\otimes(\sum_{ij}\overrightarrow{m}_i\otimes\overrightarrow{like}_{ij}\otimes\overrightarrow{d}_j)\otimes\overrightarrow{d}_4\right)\\
&= h\left(\sum_{ijkl}\langle\overrightarrow{m}_1\mid\overrightarrow{m}_l\rangle\,\overrightarrow{does}\,\langle\overrightarrow{m}_l\mid\overrightarrow{m}_k\rangle\,\overrightarrow{not}\,\langle\overrightarrow{m}_k\mid\overrightarrow{m}_i\rangle\,\overrightarrow{like}_{ij}\langle\overrightarrow{d}_j\mid\overrightarrow{d}_4\rangle\right)\\
&= h\left(\sum_{ijkl}\delta_{1l}\,\overrightarrow{does}\,\delta_{lk}\,\overrightarrow{not}\,\delta_{ki}\,\overrightarrow{like}_{ij}\delta_{j4}\right)\\
&= h\left(\overrightarrow{does}\otimes\overrightarrow{not}\otimes\overrightarrow{like}_{14}\right)\\
&= h\left((|00\rangle+|11\rangle)\otimes(|10\rangle+|01\rangle)\otimes\overrightarrow{like}_{14}\right)\\
&= h\left(|0010\overrightarrow{like}_{14}\rangle+|0001\overrightarrow{like}_{14}\rangle+|1110\overrightarrow{like}_{14}\rangle+|1101\overrightarrow{like}_{14}\rangle\right)\\
&= |0\rangle\langle 0\mid 1\rangle\langle 0\mid\overrightarrow{like}_{14}\rangle+|0\rangle\langle 0\mid 0\rangle\langle 1\mid\overrightarrow{like}_{14}\rangle+\\
&\quad |1\rangle\langle 1\mid 1\rangle\langle 0\mid\overrightarrow{like}_{14}\rangle+|1\rangle\langle 1\mid 0\rangle\langle 1\mid\overrightarrow{like}_{14}\rangle\\
&= |0\rangle\langle 1\mid\overrightarrow{like}_{14}\rangle+|1\rangle\langle 0\mid\overrightarrow{like}_{14}\rangle\\
&= \begin{cases}|1\rangle, & \text{if } \overrightarrow{like}_{14}=|0\rangle.\\ |0\rangle, & \text{if } \overrightarrow{like}_{14}=|1\rangle.\end{cases}
\end{aligned}
$$

Hence, the meaning of *Bob does not like Diet-Coke* is true, if $\overrightarrow{like}_{14}$ is false, i.e. if the meaning of *Bob likes Diet-Coke* is false."

A main distinction for the implementation of the logical NOT-gate is given by the syntax in terms of predicative versus non-predicative NOT.

Definition 3.3.1. *A NOT that acts on a verb is defined in linguistics to be* predicative *when it is used in a predicate noun phrase (predicate NP) following a linking verb, such as usually* to be*, and a noun that limits or defines the subject of the main clause. Otherwise, such a NOT is defined as* non-predicative*. For details and examples see [9].*

Example 3.3.2. *For the example treated beforehand such a distinction can be formulated as:*

Bob does not like Diet-Coke. *versus:* *It is not the case that Bob likes Diet-Coke.*

3.4 Higher Dimension

First one may think of generalizing the negation as modelled by the two-dimensional Pauli X-gate to the n-dimensional case corresponding to the matrix with all 1s on the off-diagonal and all other entries equal to 0. This matrix is obviously involutary for any dimension and thereby also allows to cover for Boolean cancelling out in the case of double negation as will be discussed later. Moreover, it does even allow to send each state in the truth-space to its opposite, which fulfils the satisfied conditions for the non-predicative NOT as long as all the entries each have one unique, semantically antisymmetric partner as opponents in the truth-state space. However, often this condition may not be given, such as in a 3-dimensional truth-space for the example sentence discussed before. In fact this does not hold in any odd-dimensional truth-space or truth-space that does not have all entries as antisymmetric pairs in meaning. This will be demonstrated in detail in the following section.

Therefore, in a first step, the predicative and non-predicative NOT for higher-dimensional truth sentence space was worked out. For the non-predicative NOT implementation the supervisors suggested in a presentation: "On a categorical approach to distributed meaning" to use a linear map as in the following definition.

Definition 3.4.1. *Define the linear map A_{NOT} for a 3-dimensional implementation of the non-predicative NOT as:*

$$A_{NOT} = \begin{pmatrix} 0 & - & 1 \\ 0 & - & 0 \\ 1 & - & 0 \end{pmatrix},$$

where the values in the second column are free parameters.

Definition 3.4.2. *In the 3-dimensional case using the same vector spaces V and W for the given grammatical sentence, define:*

$$\overrightarrow{s}_{ij} = \begin{cases} \overrightarrow{e}_1, & \text{if } m_i \text{ likes } d_j. \\ \overrightarrow{e}_2, & \text{if } m_i \text{ is indifferent to } d_j. \\ \overrightarrow{e}_3, & \text{if } m_i \text{ dislikes } d_j. \end{cases} \qquad \text{with } \overrightarrow{e}_k = \begin{pmatrix} 0 \\ k=1 \\ 0 \end{pmatrix}, \quad \text{for row k = 1,2,3.}$$

If we want to model a NOT-gate for the 3-dimensional truth-space derived from this $\overrightarrow{s}_{ij}$, then the simple generalization of the Pauli X-gate does not work. This is due to the fact that from the linguistic and logical interpretation the negation of the indifferent state should be either of the other two states as they stand for a fixed, strong opinion. That means the indifferent state does not have

a unique semantically antisymmetric partner. Instead, it has two of them $\overrightarrow{e}_1$ and $\overrightarrow{e}_3$. This fixes the values of the second column of A_{NOT} as well.

Theorem 3.4.3. *To model the non-predicative NOT-gate for a 3-dimensional truth-space derived from $\overrightarrow{s}_{ij}$ as defined in 3.4.2, including the indifferent state, the following linear map applies:*

$$A_{NOT} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

Proof. A_{NOT} applied to $\overrightarrow{e}_1$ or $\overrightarrow{e}_3$ returns the desired opposite, as fixed in 3.4.1. For $\overrightarrow{e}_2$:

$$A_{NOT} \cdot \overrightarrow{e}_2 = \overrightarrow{e}_1 + \overrightarrow{e}_3$$

Hence, implementing this linear map gives the desired result of mapping the indifferent state to either of the two others in the 3-dimensional case. □

This non-predicative version of NOT can also be expanded to the n-dimensional case in the same manner, if one considers further degrees of attitude or degrees of truth in the n-dimensional truth-space that for instance, are antisymmetric pairs. This is demonstrated in the example below.

Example 3.4.4. *Set up a 5-dimensional version of $\overrightarrow{s}_{ij}$ by considering an example of a 5-dimensional vector with each entry from the standard basis. This represents a logical disjunction of the five states in the truth-space. Here $\overrightarrow{love}$ is the semantically antisymmetric partner to $\overrightarrow{hate}$, and similarly $\overrightarrow{like}$ to $\overrightarrow{dislike}$. This means each entry in any row j is opposite in meaning to its unique partner state at $n - j + 1$, except for the indifferent state in the middle that has no such unique partner state.*

$$\begin{pmatrix} \overrightarrow{love} \\ \overrightarrow{like} \\ \overrightarrow{feel\ indifferent\ to} \\ \overrightarrow{dislike} \\ \overrightarrow{hate} \end{pmatrix}$$

Alternatively, one can combine these with adverbs to cover the "degrees of quantity" of the verb under consideration, such as for the same verb $\overrightarrow{like}$:

$$\begin{pmatrix} \overrightarrow{like\ extremely} \\ \overrightarrow{quite\ like} \\ \overrightarrow{like} \\ \overrightarrow{quite\ not\ like} \\ \overrightarrow{extremely\ not\ like} \end{pmatrix}$$

Theorem 3.4.5. *If one includes the state of being indifferent without any antisymmetric partner for it, but all the other states having their antisymmetric partner, then n will always be odd for this scenario and the linear map for the logical NOT-gate can then be implemented as follows:*

$$A_{NOT_{n-dim}} = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

Theorem 3.4.6. *For the predicative NOT in the 3-dimensional case with the same indifferent state in $\overrightarrow{s}_{ij}$ one can use the following linear map:*

$$P_{NOT} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Proof.

$$\begin{aligned} P_{NOT}(\overrightarrow{e}_1) &= \overrightarrow{e}_2 + \overrightarrow{e}_3 \\ P_{NOT}(\overrightarrow{e}_2) &= \overrightarrow{e}_1 + \overrightarrow{e}_3 \\ P_{NOT}(\overrightarrow{e}_3) &= \overrightarrow{e}_1 + \overrightarrow{e}_2 \end{aligned}$$

□

Definition 3.4.7. *Define the matrix J as the square matrix with all entries equal to 1.*

$$\rightarrow (P_{NOT})^2 = I_{3\times 3} + J_{3\times 3}.$$

Remark: Note that P_{NOT} is not involutary. Double negation would correspond to a uniform shift by 1 of each state in the meaning vector of the associated verb.

Theorem 3.4.8. *The 3-dimensional predicative NOT-gate can be expanded to an n-dimensional predicative version in the same way. One linear map for an n-dimensional predicative NOT-gate under these conditions is* $P_{NOT_{n-dim}} = -I_{n\times n} + J_{n\times n}$*, explicitly as follows:*

$$P_{NOT_{n-dim}} = \begin{pmatrix} 0 & 1 & 1 & \cdots & \cdots & \cdots & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & \cdots & \cdots & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & \cdots & \cdots & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & \cdots & \cdots & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & \cdots & \cdots & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & \cdots & \cdots & \cdots & 1 & 1 & 0 \end{pmatrix}$$

Remark: The expansion even holds generally for any $n \in \mathbb{N}$, and for any degree of quantity of the verb. Even the assumption of having symmetric or antisymmetric pairs in the truth-space is not required for this map to hold. This allows e.g. to cover for a grammatical sentence, such as Bob prefers or does not prefer . . . , as a further degree of like. Furthermore, this is unique in respect to a given basis (standard basis here) only depending on the scaling, e.g for linguistic emphasis, of the vectors in the truth-space. Hence, this yields a linear map that can be implemented very generally in the framework.

However, when one thinks of the general semantics of negation of sentences, one can see that even the non-predicative NOT does not necessarily need to oppose the meaning to a unique or a few partner states. In many interpretations it is grasped as yielding any of the other states in the truth-space as well, as outlined in the example below.

Example 3.4.9. *Bob does not like Diet-Coke* $\cong$ *Bob loves Diet-Coke.* OR *Bob is indifferent to Diet-Coke.* OR *Bob dislikes Diet-Coke.* OR *Bob hates Diet-Coke.* OR . . .

This also holds vice-versa for predicative NOT. Hence, noting this difficulty and relative inaccurate interpretation of the language meaning in this distinction, advice by Professor Pulman was followed and then the corresponding worked out under some of the modern philosophical and logical interpretations of negation as in [8].

3.5 Modern Negation

Professor Horn's book on the logic of negation [8] states that the main difference between predicative and non-predicative NOT is in one sense just that the predicative NOT makes the statement a little weaker compared to the predicative use, which in the vector space and pregroup framework may be modelled by a multiplication with a certain factor. However, it also points out the main opposite ideas of the conception of negation. It starts with the initial and very distinct philosophical ideas of Aristotle's versus Plato's. Aristotle's opinion as expressed in his: *Categories*, is that by using NOT with a verb, we mean the **contradictory** (Affirmative to Negative) and thereby the actual opposite, or mathematically orthogonal or negative of the statement. This corresponds to the interpretation of the first n-dimensional matrix that I set up when looking into the picture of non-predicative NOT. On the other hand, Plato's view is very different and explained in an extract in his rebuttal:

> When we say not-being, we speak, I think, not of something that is the opposite of being, but only of something different.

This is a so called **contrary** view for the conception of negation, and corresponds to what Professor Pulman pointed out and thereby to the matrix set up beforehand for the predicative NOT back then, where negation is to make all other states in the n-dimensional space possible. From a logical point of view the main distinction between these two philosophical views on the interpretation of negation lies in the law of contradiction (LC) and the law of excluded middle (LEM). As shown in [8]: "LC holds for both contradictory as well as contrary views, but LEM only for contradictory."

LC: $\neg(\mathbf{p} \wedge \neg\mathbf{p})$
LEM: $\mathbf{p} \vee \neg\mathbf{p}$

Semantically this is expressed as:
LC: Two contradictory sentences are not true together
LEM: Two contradictory sentences are not false together

In the quantified, term-based version:
LC_{term} : $\neg\exists\mathbf{x}(\mathbf{Px} \wedge \neg\mathbf{Px})$
LEM_{term} : $\forall\mathbf{x}(\mathbf{Px} \vee \neg\mathbf{Px})$

For the multiple reasons argued beforehand, predicative versus non-predicative position of NOT does not really have a meaningful influence on the semantics except for a strengthening

or weakening, respectively. In this sense, I think it is more sensible to consider this distinction that was and in many ways still is the main logical distinction for the NOT according to the book. The weakening of the predicative position can then just be undertaken by a multiplication with a constant lambda or similar, as was the idea of Dr. Coecke for a similar case when $\overrightarrow{does}$ is used to emphasize. Another advantage of this picture of philosophical or logical distinction is that it may be linguistically universal, meaning the idea applies to all natural languages, at least for grammatical sentences and when we consider the NOT applied to a verb, from my thinking in all of where I can judge: English, German, French, Spanish, Italian, Swiss, Latin, Greek and also Chinese, since it is about the conceptual philosophical distinction. There are some special cases such as the French: "j'espere que non" versus "je n'espere pas que" of contrary versus contradictory that flow together in e.g. their German counterpart: "Ich hoffe nicht", which can be interpreted as both of the concepts. The choice will mostly depend on the context and thereby not present a problem.

For the case of double negation from a propositional logic point of view, naturally the negation should cancel out in both non-predicative as well as predicative use of NOT (Duplex Negatio Affirmat). For the Stoics (following Plato's view) this is represented by the Law of Double Negation:

$$\frac{\neg\neg\overline{\mathbf{P}}}{\therefore \overline{\mathbf{P}}} \qquad (LDN)$$

Let us add LDN to e.g an intuitionistic logic concept giving a sense of a linear logic and let the entries in the truth-space of even dimension be symmetric. Consequently, any linear map for the logical NOT-gate in the interpretation of contrariety as seen by the Stoics would have to be involutary, meaning square roots of the n-dimensional square matrix and so have to satisfy all of the following properties simultaneously.

$$\text{i.)} \qquad \overline{P} = \sqrt{I_{n\times n}} \rightarrow det(\overline{P}) = \pm 1$$

$$\text{ii.) till n+i.)} \qquad \overline{P} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ j=1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = [k] \begin{pmatrix} 1 \\ \vdots \\ 1 \\ j=0 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

The factor k in the conditions stands for a possible renormalization in the end as is often the case in quantum computer science as well as information retrieval. However, this does not change the theorem and proof to hold as shown below, since it corresponds just to an affine transformation of all the values; nor does any possible linear shift associated to emphasis etc., since that will take place on both sides of the equality. There are many papers from the 50s and 60s investigating such general square roots of the identity matrices and many of the additional properties can be satisfied in a linear map as was searched and found using Matlab and Mathematica. However, it turns out that there is no such linear map satisfying all these properties as is proven next.

Theorem 3.5.1. *There exists no linear map for such a logical NOT-gate satisfying the properties of Plato's logic that can satisfy all these n+1 distinct conditions, including LDN.*

Proof.

$$\begin{aligned} \text{From ii.)} \quad \overline{P} &= (J - I) \\ \rightarrow \quad \overline{P}^2 &= (J - I)^2 = J^2 - 2IJ + I = (n-2)J_{n\times n} + I_{n\times n} \\ &\neq I_{n\times n}\,. \end{aligned}$$

□

Alternatively, one could also allow factors λ_i for non-polar terms. Analysing and proving for instance the 4-dim. case (analogously for higher dimensions) the following adjusted properties would have to be satisfied in addition to the matrix being involutary:

$$\text{ii.) } \overline{P}\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \lambda \\ \lambda \\ 1 \end{pmatrix} \qquad \text{iii.) } \overline{P}\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \lambda \\ 0 \\ 1 \\ \lambda \end{pmatrix}$$

$$\text{iv.) } \overline{P}\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \lambda \\ 1 \\ 0 \\ \lambda \end{pmatrix} \qquad \text{v.) } \overline{P}\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ \lambda \\ \lambda \\ 0 \end{pmatrix}$$

Proof. From these properties that have to be satisfied it follows that:

$$\overline{P} = \begin{pmatrix} 0 & \lambda & \lambda & 1 \\ \lambda & 0 & 1 & \lambda \\ \lambda & 1 & 0 & \lambda \\ 1 & \lambda & \lambda & 0 \end{pmatrix} \quad \rightarrow \quad \overline{P}^2 = \begin{pmatrix} 2\lambda^2+1 & 2\lambda & 2\lambda & 2\lambda^2 \\ 2\lambda & 2\lambda^2+1 & 2\lambda^2 & 2\lambda \\ 2\lambda & 2\lambda^2 & 2\lambda^2+1 & 2\lambda \\ 2\lambda^2 & 2\lambda & 2\lambda & 2\lambda^2+1 \end{pmatrix} \neq I_{n\times n},$$

since the λs cannot be 0 for the properties to hold and give the desired interpretation of the NOT-gate as being contrary. □

Aristotle on the other hand, introduced contradictory negation in such a way as to be incapable of applying to its own output, hence LDN is not included in the term logic of Aristotelians. Therefore, for this philosophical interpretation the matrix A_{NOT} as set up before can in fact be implemented. Aristotle may not have been aware of linear maps in the modern sense back then, but he clearly recognized the difficulty from a logical and linguistic point of view. As a matter of fact it does turn out that there is no such linear map either. Such a map for the logical NOT-gate would have to fulfil all of the following $n+2$ conditions, plus possibly further properties that may be relevant:

i.) $\overline{A} = \sqrt{I_{n\times n}} \rightarrow det(\overline{A}) = \pm 1$

ii.) For the indifferent state: $\overline{A} \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 0 \\ \vdots \\ 1 \end{pmatrix}$

iii.) till n+ii.) For the indifferent state: $\overline{A} \begin{pmatrix} 0 \\ \vdots \\ j=1 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \vdots \\ (n-j+1)=1 \\ \vdots \\ 0 \end{pmatrix}$

Theorem 3.5.2. *There is no linear map for such a logical NOT-gate satisfying the properties of Aristotle's logic that can satisfy all these n+2 distinct conditions, including LDN.*

Proof. In order to satisfy property ii.), one has:

$$\overline{A}^2 \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} = \overline{A} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} | & | & | \\ \overline{a}_1 & \overline{a}_2 & \overline{a}_3 \\ | & | & | \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} | \\ \overline{a}_1 \\ | \end{pmatrix} + \begin{pmatrix} | \\ \overline{a}_3 \\ | \end{pmatrix}$$

$$\stackrel{!}{=} k \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = k \begin{pmatrix} | \\ \overline{a}_2 \\ | \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} | \\ \overline{a}_1 \\ | \end{pmatrix} + \begin{pmatrix} | \\ \overline{a}_3 \\ | \end{pmatrix} \stackrel{!}{=} k \begin{pmatrix} | \\ \overline{a}_2 \\ | \end{pmatrix}$$

$$\rightarrow det(\overline{A}) = 0$$

This is a contradiction to i.) and, therefore, no such linear map exists. Again any possible renormalization factor k does not change the proof to hold as shown above, since the second column will still be a linear combination of the other two; nor does any possible linear shift associated to emphasis etc. because that will take place on both sides of the equality. □

However, even though the two theorems 3.5.1, 3.5.2 and their proofs show that there are no non-linear maps satisfying all the required properties, one can set up outputs of non-linear maps that do the job. For each case of input independently they are actually linear, and in fact satisfy all the $n + 1$ required properties for the former and all the $n + 2$ required properties for the latter philosophical interpretation of the logical NOT-gate.

Definition 3.5.3. *Define the non-linear outputs of such maps $A(v)$ and $P(v)$ applied to an input vector v as follows:*

$$A(v) := \begin{cases} A_{NOT_{n-dim}} \cdot v, & \textit{if } v = \overrightarrow{e}_i, \textit{ for } i \in \{1, \ldots, n\}. \\ \overrightarrow{e}_i, & \textit{otherwise } v = A_{NOT_{n-dim}} \cdot \overrightarrow{e}_i. \end{cases}$$

$$P(v) := \begin{cases} P_{NOT_{n-dim}} \cdot v, & \textit{if } v = \overrightarrow{e}_i, \textit{ for } i \in \{1, \ldots, n\}. \\ \overrightarrow{e}_i, & \textit{otherwise } v = P_{NOT_{n-dim}} \cdot \overrightarrow{e}_i. \end{cases}$$

Then these maps can be used for the logical NOT-gate in any higher dimensions and also cover for cancelling out in the case of double negation corresponding to the requirements of propositional logic or Plato's view of double negation. This can be implemented, since the only times that $A_{NOT_{n-dim}}$ or $P_{NOT_{n-dim}}$ cannot be applied directly with LDN as an axiom is when they are applied to already negated vectors that are not of the form $\overrightarrow{e}_i$ for $i \in \{1, \ldots, n\}$. Hence, they can all be distinguished from the case when the matrices can be applied directly and so be implemented.

According to Jaakko Hintikka ([8], 1968), a leading Finnish philosopher and logician:

> while a proposition or predicate may be logically equivalent to its double negation, the doubly negated form tends to convey one of the three "residual meanings":
>
> 1.) to indicate hesitation or uncertainty
> 2.) to signal diffidence
> 3.) to express irony

This will also depend on the context. We may not want the matrices to be involutary, but possibly yield some free variables in relation to the context or certain categories of verbs that it is referring to, depending on the text and implementation. Evidently, in this very recent picture, compared to the classical two opposite philosophical ideas of negation in linguistics, LDN does not apply either. It may model the semantics of languages more accurately. As a matter of fact many of the modern views agree or incorporate this picture of the semantics of negation. Since LDN does not apply, one can actually implement the n-dim. matrices and thereby linear maps $A_{NOT_{n-dim}}$ and $P_{NOT_{n-dim}}$ that were set up in section 3.4 on higher dimension. The square of the second matrix (predicative or contrary view) yields the n-dim. identity matrix as desired and in addition $n \cdot J$, so that in fact similar to a possible indication of emphasis by a constant factor, this adds the same value to every state of the output in the truth-space, as one would want when modelling by the same means the additional irony or hesitation that is expressed.

Chapter 4

Orthogonal Negation

Alternatively, one can approach the problem of setting up such a logical NOT-gate as mentioned before by using projections to orthogonal subspaces. Dominic Widdows, former member of the faculty at Oxford's Computing Laboratory, presents an important theorem in [17] that allows to implement this approach and from which also a version for concatenation of several logical NOTs or ANDs can be set up.

He points out that standard information retrieval systems when processing an input, such as A NOT B to return all the results that include A, but not B are not very efficient. This is because the way they usually work is by searching the whole space Ω for A, and then in a second step taking the $A \subset \Omega$ and look for any B in it that are then cancelled out. This means that the whole search procedure has to be processed essentially twice. In his paper during his research at Stanford University Dominic Widdows describes a method using projections onto orthogonal subspaces that allows to remove unwanted terms directly from the initial query. In addition, it even allows to not only remove unwanted strings, but unwanted meaning. "Compared to standard Boolean methods this reduces the occurrence of synonyms and neighbours by as much as 76%," according to his paper [17] that will be applied in the following.

4.1 Orthogonal Negation Method

This orthogonal negation method is also based on a vector model and uses the same vector negation operator as in quantum logic. The main idea is to use orthogonality in the sense of the following definition in order to model irrelevance in vector spaces.

Definition 4.1.1. *Two words a and b are considered irrelevant to one another, if their vectors are orthogonal. This means that a and b are mutually irrelevant, if $a \cdot b = 0$.*

Definition 4.1.2. *Let $a, b \in V$ and $A, B \leq V$. By NOT A we mean $A^{\perp}$ and by NOT a, we mean $\langle a \rangle^{\perp}$, where $\langle a \rangle = \{\lambda a : \lambda \in \mathbb{R}\}$ is the 1-dimensional subspace generated by a. By a NOT B we mean the projection of a onto $B^{\perp}$ and by a NOT b we mean the projection of a onto $\langle b \rangle^{\perp}$.*

The following demonstrates how to use these notions to perform calculations with individual term or query vectors in a form which is simple to program and efficient to run.

Theorem 4.1.3. *Let $a, b \in V$. Then a NOT b is represented by the vector*

$$a \textit{ NOT } b \equiv a - \frac{a \cdot b}{|b|^2} b, \tag{4.1}$$

where $|b|^2 = b \cdot b$ is the modulus of b.

Proof. One can give a straightforward proof by applying the trick suggested from definition 4.1.1 as in [18] of taking the scalar product with b on both sides of the equality:

$$(a \textit{ NOT } b) \cdot b = (a - \frac{a \cdot b}{|b|^2} b) \cdot b = a \cdot b - \frac{(a \cdot b) \cdot (b \cdot b)}{|b|^2} = a \cdot b - \frac{(a \cdot b) \cdot (b \cdot b)}{(b \cdot b)}$$

$$= 0$$

$$\rightarrow (a \textit{ NOT } b) \perp b.$$

Hence, from definition 4.1.1, it follows that a NOT b is exactly the part of a which is irrelevant to b. □

Remark: Theorem 4.1.3 takes a particularly simple form for normalized vectors:

$$a \textit{ NOT } b = a - (a \cdot b) b, \tag{4.2}$$

which in practice is then renormalized for consistency just like in quantum physics.

One computational benefit is that theorem 4.1.3 gives a single vector for a NOT b, so finding the similarity between any other vector and a NOT b is just a single scalar product computation. Disjunction is also simple to envisage, the expression b_1 OR ... OR b_n being modelled by the subspace $B = \{\lambda_1 b_1 + \ldots + \lambda_n b_n : \lambda_i \in \mathbb{R}\}$. Theoretical motivation for this formulation can be found in (Birkhoff and von Neumann, 1936, 1, 6) and [18]: for example, B is the smallest *subspace* of V which contains the set $\{b_j\}$. Computing the similarity between a vector a and this subspace B is computationally more expensive than for the negation of theorem 4.1.3 because the scalar product of a with (up to) n vectors in an orthogonal basis for B must be computed.

4.1.1 Higher Dimension

The gain we get by comparing each document with the query a NOT b using only one scalar product operation is absent for disjunction. However, this benefit is regained in the case of *negated* disjunction. Suppose we negate not only one argument, but several. If a user specifies that they want documents related to a *but NOT* $b_1, b_2, \ldots, b_n$, then (unless otherwise stated) it is clear that they only want documents related to *none* of the unwanted terms b_i (rather than, say, the average of these terms).

This motivates a process which can be thought of as a vector formulation of the classical de Morgan equivalence $\neg a \wedge \neg b \equiv \neg(a \vee b)$, by which the expression:

$$a \textit{ AND NOT } b_1 \textit{ AND NOT } b_2 \ldots \textit{ AND NOT } b_n.$$

is translated to:

$$a \textit{ NOT } (b_1 \textit{ OR } \ldots \textit{ OR } b_n). \tag{4.3}$$

Using Definition 4.1.2, this expression can be modelled with a unique vector which is orthogonal to all of the unwanted arguments $\{b_1\}$. However, unless the vectors $b_1, \ldots, b_n$ are orthogonal (or identical), we need to obtain an orthogonal basis for the subspace b_1 *OR* ... *OR* b_n, before we can implement a higher-dimensional version of Theorem 4.1.3. This is because the projection operators involved are in general non-commutative, one of the hallmark differences between Boolean and quantum logic. In this way vector negation generates a meaning vector which takes into account the similarities and differences between the negative terms.

A query for

chip NOT computer, silicon

is treated differently from a query for

chip NOT computer, potato.

Vector negation is capable of realising that for the first query, the two negative terms are referring to the same general topic area, but in the second case the task is to remove radically different meanings from the query. This technique has been used to remove several meanings from a query iteratively, allowing a user to 'home in' on the desired meaning by systematically pruning away unwanted features.

Theorem 4.1.4. *In order to derive an orthogonal basis for the subspace b_1 OR ... OR b_n, one can apply the Gram-Schmidt orthogonalization process on any given basis. Normalization in each step of the process even allows to get an orthonormalized basis so that the simpler version of the theorem above eq. 4.2 can be applied for a higher-dimensional implementation of the result.*

4.1.2 Implementation in Computational Linguistics

Example 4.1.5. *One can use this approach of orthogonal projections to subspaces and apply it to the examples discussed beforehand e.g. in the 5 or 6-dimensional example. Let V be the vector space spanned by the meanings of all the verbs or a possible subset thereof, if one wants to apply the model to a specific context. Then set up the following:*

$$\begin{aligned} a = \overrightarrow{does}, \quad & \text{and A the associated vector subspace of V.} \\ b \in B, \quad & \text{and B } \leq V. \end{aligned}$$

Thereby the subspace B may be spanned by the following corresponding to the standard basis:

$$B = \{\overrightarrow{love}, \overrightarrow{like}, \overrightarrow{feel\ indifferent\ to}, \overrightarrow{dislike}, \overrightarrow{hate}\}.$$

Then one can either apply the general theorem using eq. 4.1 for the case when one is also having linear combinations of the meanings in the vector space, or one can use the simpler eq. 4.2, when one is only dealing with the states outlined before that are given by the vectors of the standard basis and thereby already fulfil the condition of being normalized.

$$\rightarrow aNOTb = \overrightarrow{does}\ \textit{NOT}\ b.$$

W.l.o.g. assign the meaning vector such that: $a \neq b$, *and e.g. let* $b = \overrightarrow{like,}$ *then:*

$$\rightarrow aNOTb = \overrightarrow{does}\ \textit{NOT}\ \overrightarrow{like} = a - \lambda b, \qquad \textit{for some scalar } \lambda.$$

This corresponds to an expression that is a linear combination of:

$$\overrightarrow{does}\ \textit{and some}\ \overrightarrow{v} \in V \backslash \{b\},$$

which is orthogonal to $\overrightarrow{like}$.

$=$ *(does) love ... OR (does) feel indifferent to ... OR (does) dislike ... OR (does) hate ...*

Or it can take on any linear combination of these verbs ... . Or in the case when B *is a proper vector subspace of* V*, then it can be even other verbs in the vector space* V *of all the verbs, followed by the rest of the sentence as indicated by the dots. Hence, this fast search procedure will then treat the vector* b *that is now orthogonal to the query term as irrelevant, and instead return sentences or documents containing linear combinations of the form above as desired.*

Any additional emphasis in meaning returned through $\overrightarrow{does}$ *is automatically handled by the renormalization that takes place in practice as stated by Widdows.*

This is exactly what is wanted for the interpretation of NOT as being contrary as seen by Plato, since only the "orthogonal" or irrelevant verb meanings in the vector space are cancelled out for any dimension of the meaning space.

For Aristotle's conception of negation this *single* NOT-version only works up to the two-dimensional case, since through this NOT all the other states or verb actions become possible except for the initial unwanted that is negated. However, using several NOTs as in theorem 4.1.4, one can cancel out the other states through orthogonality as desired and thereby also derive a NOT-gate that works for Aristotle's contradictory logic. With the high efficiency that was shown by Widdows in statistical analysis this can be a very efficient method for an implementation of NOT.

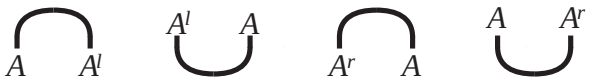
Chapter 5

Diagrammatic Calculus

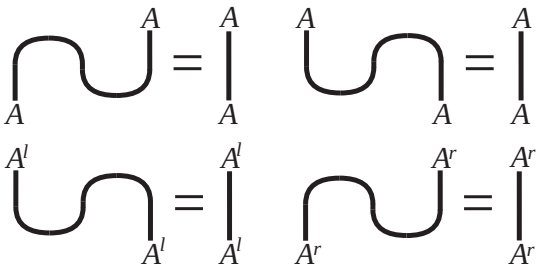
5.1 Diagrammatic Calculus of Compact Closed Categories

5.1.1 Categories

Definition 5.1.1 (Compact Closed Categories)**.** *A monoidal category is defined in [6] to be compact closed in the diagrammatic calculus, if for each object A there are also objects A^r and A^l and the following for the category morphisms $\eta^l, \epsilon^l, \eta^r, \epsilon^r$, respectively:*



that fulfil the axioms:



In the main paper that is expanded here [6] it is shown that vector spaces, linear maps as well as the tensor product form such a compact closed category. Furthermore, it then also shows that even pregroups form a compact closed category as well. As a demonstration, the following shows that compact closure holds:

Proof. For pregroups the morphisms for compact closure, listed in the same order of the morphisms as before, are:

$$\eta^l = [1 \leq pp^l] \qquad \epsilon^l = [p^l p \leq 1]$$

$$\eta^r = [1 \leq p^r p] \qquad \epsilon^r = [pp^r \leq 1]$$

Juxtaposition of these types clearly reduces to identity in all cases and, hence pregroups fulfil the compact closure property too. □

The following proves the correspondence between compact closed categories and the diagrammatic calculus as expanded in [16]. It is this correspondence that allows to use the straightforward reductions in the diagrammatic calculus in order to be able to do such very efficient calculations as seen in chapter 3.

Theorem 5.1.2. *A well-typed equation between morphisms in the language of compact closed categories follows from the axioms of compact closed categories iff it is true, up to diagrammatic isomorphisms, in the diagrammatic calculus.*

Proof. This was implicitly proven in [10] by Kelly and Laplaza in 1980. □

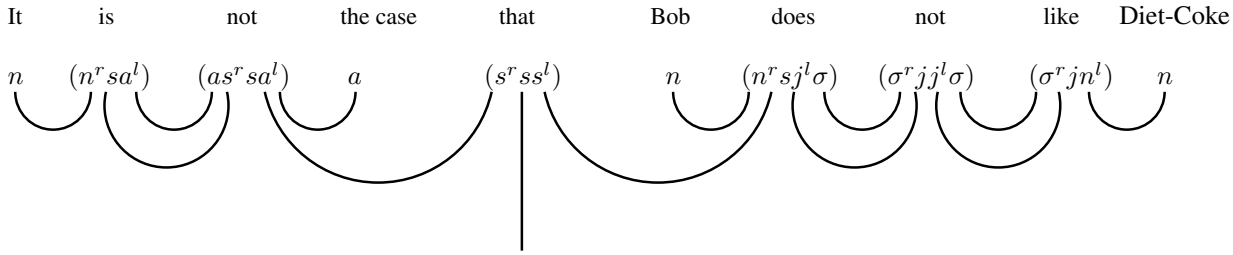
The following gives the equivalent for so called "strongly" or "dagger compact closed categories" as introduced by Samson Abramsky and supervisor Bob Coecke at Oxford University as an axiomatic framework to quantum mechanics.

Theorem 5.1.3. *A well-typed equation between morphisms in the language of strongly compact closed categories follows from the axioms of strongly compact closed categories iff it is true, up to diagrammatic isomorphisms, in the diagrammatic calculus.*

Proof. Soundness follows by induction. Completeness can then be proven using the additional axiom for a strongly or dagger compact closed category, and theorem 5.1.2 for compact closed categories as shown above following the same procedure as in the proof of Theorem 3.10 in [16]. □

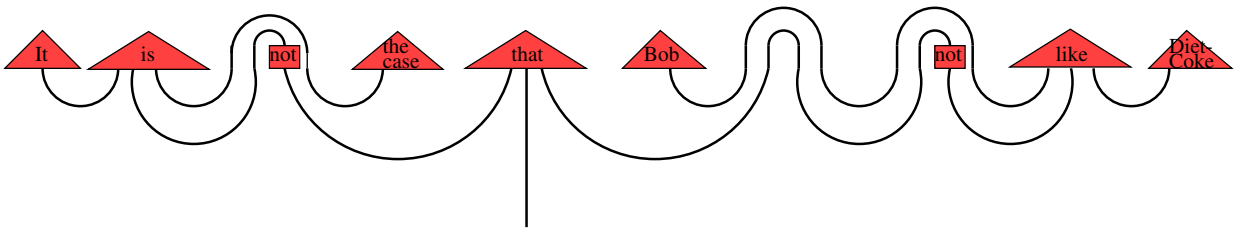
5.1.2 Syntax in Graphical Calculus

If we construct the sentence considered beforehand so that it has double negation where the first is predicative and the second is non-predicative, then one gets e.g. the following types and representations in the graphical calculus. It is to be noted that these types are exemplary and not derived from a linguistic corpus as usual in categorical treatment so that they reduce the sentence in order for it to be grammatical. Besides, they also satisfy the semantic requirements of the framework. The following first example is to show the types associated to the two versions of predicative and non-predicative $\overrightarrow{not}$ in one grammatical sentence. The two are equivalent except for a strengthening in one of them as outlined before according to [8].

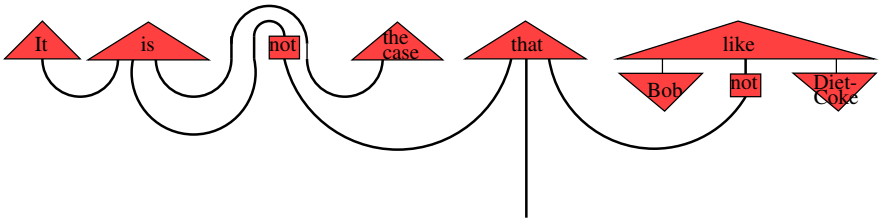


5.1.3 Semantics Implemented in Diagrammatic Calculus

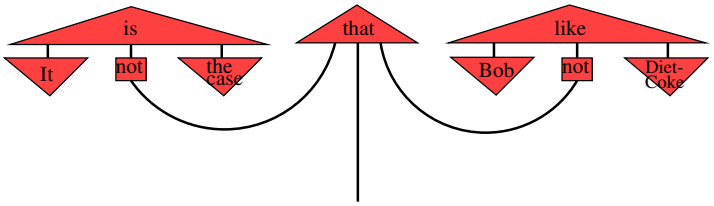
Example 5.1.4. *The next picture gives the equivalent representation in the diagrammatic calculus of compact closed categories. The arcs that indicate the type reductions in a pregroup grammar are exactly the "cups" of the compact closed structure as a corrolary from the proofs above. The triangles represent the meanings of the individual words and in the quantum analogy correspond to the states of a quantum system involved in an information protocol. The $\overrightarrow{does}$ is defined as in section 3.2 and similarly the $\overrightarrow{not}$ where for higher dimensions the matrices that were worked out in the thesis apply to the box operator.*



This can be simplified applying the reduction of the right-hand side as outlined in section 3.2. Reversed triangles indicate the corresponding functionals in the dual space or the bra-ket relation in Dirac notation.

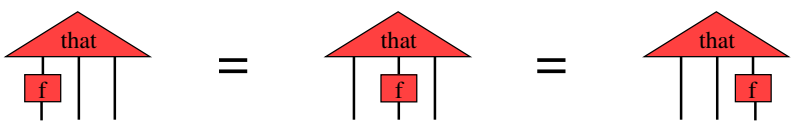


Further reduction thereof:



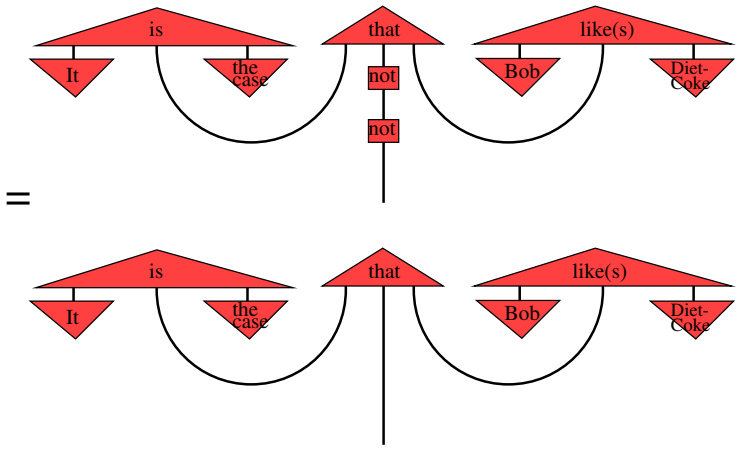
The next section will also cover the law of double negation (LDN) for Plato's or Hintikka's conception up to an emphasis. The emphasis is modelled in the vector spaces corresponding to the triangles and, hence has no influence on the diagrammatic calculus. In the example discussed the main clause is predicative and refers to the following clause without adding anything new to the meaning, except for a strengthening or weakening of the following clause. Therefore, from intuition it makes sense in this case logically and semantically when such an input applies to the $\overrightarrow{that}$ operator to consider the $\overrightarrow{not}$ boxes as double negation of the same positive statement in the clause of reference. One can then apply LDN for the $\overrightarrow{not}$ operators to cancel each other out and return the positive sentence.

Logical Interpretation I: In order to apply LDN in the diagrammatic calculus, when the main clause acts to negate the verb after the $\overrightarrow{that}$ operator, we introduce the following *logical interpretation* for $\overrightarrow{that}$ in this case, since the $\overrightarrow{that}$ functions as the connective between the clauses and so also between the various $\overrightarrow{not}$ operators.

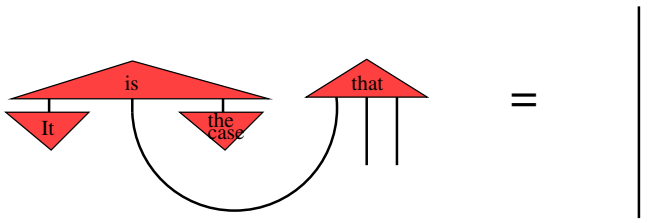


In this case the action of the $\overrightarrow{that}$ correspondes to the logical XOR-gate that implements exclusive disjunction. If the two inputs to it, corresponding to two $\overrightarrow{not}$ operators, are both false or both true, then it will return true; as wanted when including LDN. On the other hand, if only one of the inputs is false, corresponding to a negative sentence as input to the $\overrightarrow{that}$, then false is returned and thereby the single negation remains. A quantum version of the XOR-gate that operates in arbitray dimensional Hilbert spaces and is used for generalized quantum teleportation as well as state purification is presented in [1]. Hence, such an XOR-gate can be implemented for the function of the $\overrightarrow{that}$ operator and even cover the case of higher-dimensional vector spaces that were set up.

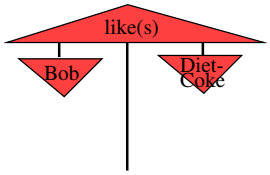
Then first use the *logical interpretation* to shift the $\overrightarrow{not}$ gates and in a second step apply LDN. As a result, the two $\overrightarrow{not}$ gates now cancel out so that further reduction is reached to the desired:



Logical Interpretation II: The clause "it is not the case that" does not add any significant meaning other than a predicative strengthening. Its subtle treatment is complex and can be part of future work. A simplification is implemented through another logical interpretation setting the sentence equal to the identity.

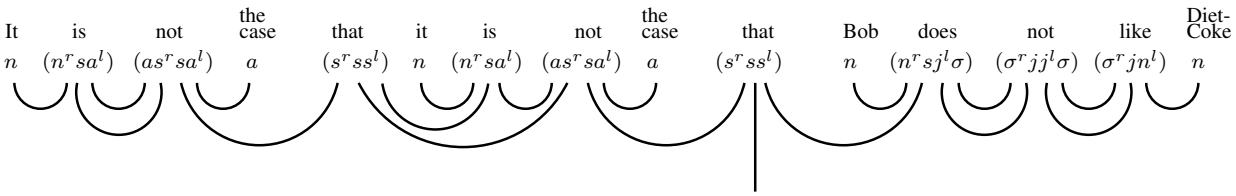


This allows to reduce the statement to the positive sentence: Bob like(s) Diet-Coke. Conjugation adaptation can be handled separately in an intermediate step or in the end. The compact representation in the diagrammatic calculus then reduces to the following final output:

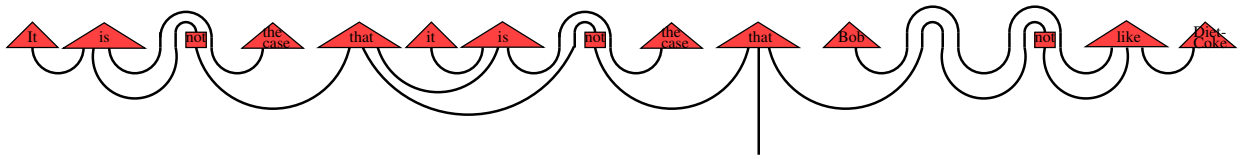


This gives the required abstractions for the higher-dimensional matrices to do the same efficient vector calculations for the meaning as in section 3.3 as an alternative to the extensive Dirac calculations.

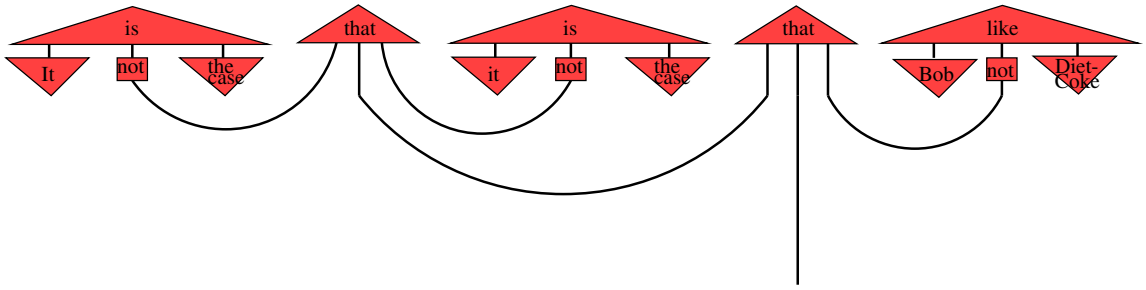
Example 5.1.5. *The next example investigates a sentence with double negation that negate the same object, followed by a third negation in form of the negated transitive sentence considered earlier. The types are the same as before and also satisfy the reduction of the sentence for it to be grammatical and the semantic properties in the category.*



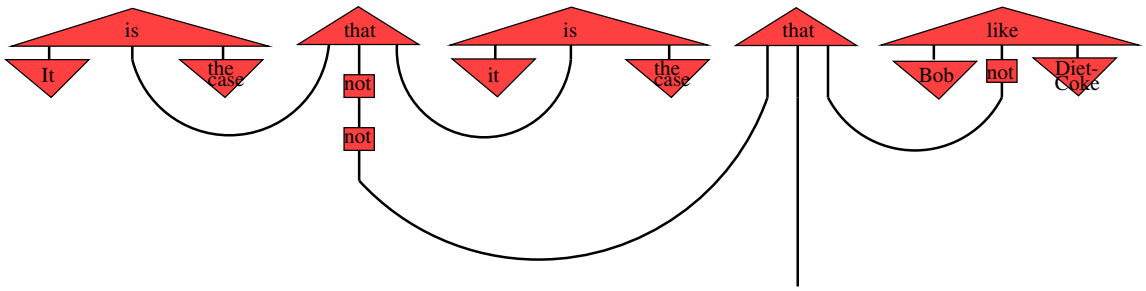
The whole picture in the diagrammatic calculus is then quite complex as follows:



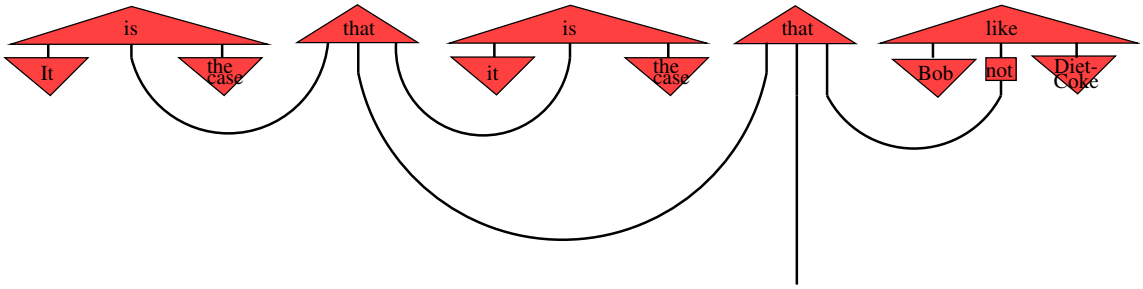
This reduces to the following:



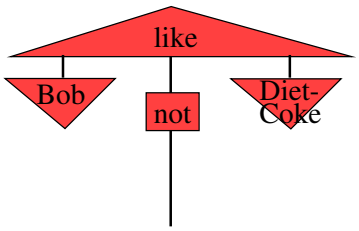
Using logical interpretation I for the $\overrightarrow{that}$, leads to the further reduction thereof:



=



Then applying twice logical interpretation II, gives the final reduction to the sentence: “Bob does not like Diet-Coke”.



Alternatively, one could also first cancel out the negations in the second $\overrightarrow{that}$. Since eventually using logical interpretation I to shift the remaining $\overrightarrow{not}$ box that is attached to the remaining $\overrightarrow{that}$ operator from the left to the right, will yield the same result.

Conclusion

A great excitement was achieved with the thesis and several distinct and promising approaches to the problem in question were followed. All of them achieve to provide such a higher-dimensional logic gate with distinct means and properties. A main focus and implementation was worked out for logical negation. Similar ideas towards the logical AND-gate and implementations in the diagrammatic calculus thereof were worked out by two other MFoCS students under the supervision of Dr. Bob Coecke.

Direct generalizations of the logical NOT-gate in the form of a swapping as in quantum computer science do not allow to cover certain desired properties such as non-symmetric (or non-antisymmetric in meaning) states in the truth-space. To be able to also handle such cases two distinct matrices as suggested by the supervisors were expanded to higher dimensions. It was recognised in the treatment that the main idea that provides the principal challenge in implementing a higher-dimensional version of such gates lies in the fulfilment of the logical and semantic properties, rather than the syntactic. From this a matrix that works uniformly for any dimension of the truth-space was explored and elaborated.

Then an important refinement of these various NOT-gates as suggested by the supervisors that are treated in modern linguistics was undertaken in accordance with logical philosophies and literature. The two main distinct philosophical conceptions of negation of the Aristotelians versus the Stoics were considered. It was discovered and proven that, if one also wants to follow the Stoics conception and cover the law of double negation (LDN) as in Boolean or Linear Logics, then there are no such linear maps that can fulfil the required properties simultaneously. Stepwise linear versions were set as outputs instead for possible implementations. It was shown that the n-dimensional matrices for any $n \in \mathbb{N}$ that were set up, adjusted appropriately, hold for the modern conception of negation with the extra meaning conveyed by double negation as defined by Hintikka.

Afterwards, an alternative approach for a mathematical implementation of such a logical NOT-gate using quantum logic was followed as suggested by David Widdows, and also expanded to higher dimensions. The method was adapted and worked out for the problem in question. It proofs very fruitful for applications in IR because irrelevance of search terms, modelled by the orthogonality of the terms trough the NOT-gate, can be handled in the initial query already. In addition, this gives a very significant benefit in the modelling of the meaning over Boolean methods.

Eventually, the higher-dimensional versions of the NOT-gate were incorporated in the diagrammatic calculus as this provides a simplification and much more efficient approach for evaluation of the meaning calculation due to the fact that vector spaces, linear maps and the tensor product all form a closed compact category. This included also approaching the challenge of double negation in the various conceptions. To conquer this an original implementation of the *that* connective was constructed using the XOR-gate (resp. the analogue quantum computer science version). This allows to reach the same diagrammatic reduction as in simpler one-negation examples that had been worked out in the recent papers by the supervisors. Moreover, specific types were associated for the complex sentences to guarantee them to be grammatical as in definition 2.3.3 and simultaneously also fulfilling the semantic property requirements in the category.

Analogue or similar methods and expansions as above can also be used to expand other logical gates or words in future work. As a generalization of the framework one can similarly work with subspaces instead of vectors. This may be less efficient in an actual implementation, but would allow to model disjunction for contrary negation by the vector sum, since that corresponds to the smallest subspace containing the two components in the sum. Then the inverse could correspond to the semantically antisymmetric partner states and linear scaling to an additional emphasis.

Moreover, future work can consist in further exploring the implementations and meaning calculations in various languages on real corpus data with the type structure given by the pregroups and generalized by the categories. At that point, one will also have to deal with complexity and efficiency of the framework and thereby optimization. It would be brilliant, if one day one can also exploit and incorporate the use of entanglement as in quantum computer science for high-efficiency of such algorithms and an automated system.

The research group at the Oxford University Computing Laboratory is holding a workshop in fall together with some of the leading experts in the field where the framework will be presented and discussed. The goal is to establish cooperation with some of the leading science universities in the United Kingdom, including the University of Sussex as well as the University of Cambridge in order to expand the framework. This may also result in implementations of this in an automated meaning translation or information retrieval system.

Bibliography

[1] G. Alber, A. Delgado et al., Generalized quantum XOR-gate for quantum teleportation and state purification in arbitrary dimensional Hilbert spaces, Abteilung für Quantenphysik, Universität Ulm, Germany, 2008.

[2] D. Bechet, A. Foret et al., Learnability of Pregroup Grammars, Studia Logica **87(2-3)**, 2007. http://hal.archives-ouvertes.fr/docs/00/19/11/12/PDF/bechet-foret-tellier.pdf

[3] W. Buszkowski, Lambek grammars based on pregroups, Logical Aspects of Computational Linguistics, 2001.

[4] C. Casadio and M. Sadrzadeh, Clitic Movement in Pregroup Grammar a Cross-linguistic Approach, Oxford University, 2010. http://www.comlab.ox.ac.uk/files/2953/CliticMovement.pdf

[5] S. Clark, B. Coecke and M. Sadrzadeh, A Compositional Distributional Model of Meaning, QI, College Publications, University of Oxford, 2008.

[6] S. Clark, B. Coecke and M. Sadrzadeh, Mathematical Foundations for a Compositional Distributional Model of Meaning, Linguistic Analysis **36** (Lambek Festschrift), 2010. http://arxiv.org/abs/1003.4394

[7] S. Clark and S. Pulman, Combining symbolic and distributional models of meaning, AAAI Press, Proceedings of AAAI Spring Symposium on Quantum Interaction, 2007.

[8] L. Horn, A Natural History of Negation, Yale University, 1989.

[9] R. Huddleston, English Grammar: An Outline, Cambridge University Press, 1988.

[10] G. M. Kelly and M. L. Laplaza, Coherence for Compact Closed Categories, Journal of Pure and Applied Algebra **19**, 1980.

[11] J. Lambek, From Word to Sentence, A Computational Algebraic Approach to Grammar, Polimetrica, Monza (MI), 2008.

[12] J. Lambek, The Mathematics of Sentence Structure, American Mathematics Monthly **65**, 154169, 1958.

[13] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.

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

[15] M. Sadrzadeh, Pregroup Analysis of Persian Sentences, University Paris-Diderot, Paris 7.

[16] P. Selinger, Dagger compact closed categories and completely positive maps, Electronic Notes in Theoretical Computer Science **170**, 139–163, 2007.

[17] D. Widdows, Orthogonal negation in vector spaces for modelling word-meanings and document retrieval, 41st Annual Meeting of the Association for Computational Linguistics, Japan, 2003.

[18] D. Widdows and S. Peters, Word Vectors and Quantum Logic: Experiments with negation and disjunction, Stanford University, 2003.