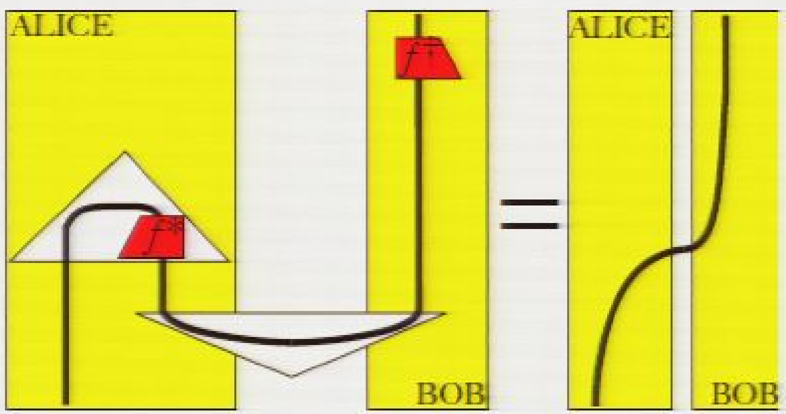
Title: A Categorical Approach to Distributed Meaning

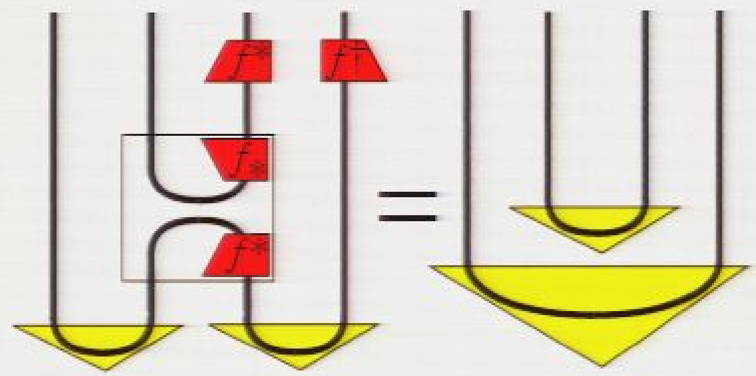
Date: Jun 02, 2009 12:15 PM

URL: http://pirsa.org/09060018

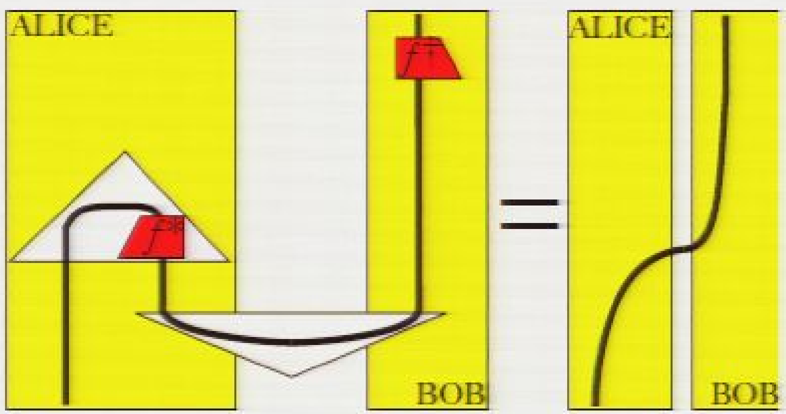
Abstract: TBA

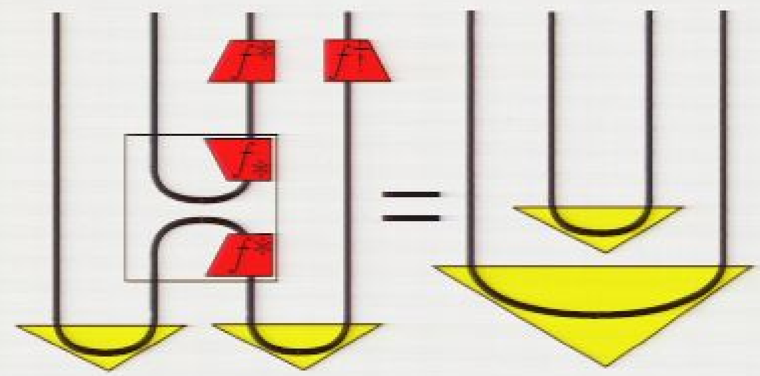
String diagrams in Quantum Protocols



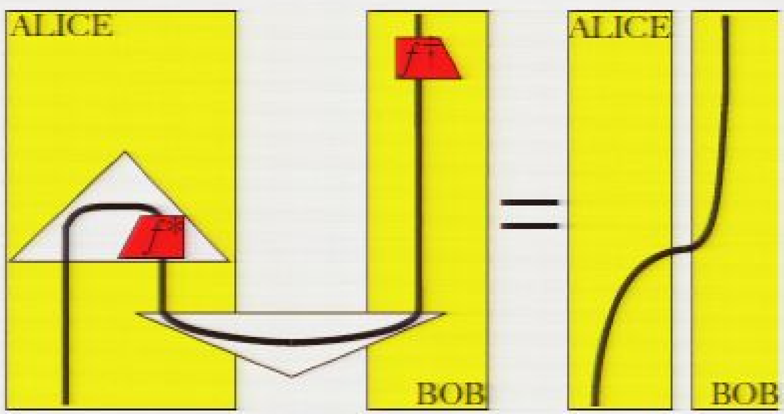


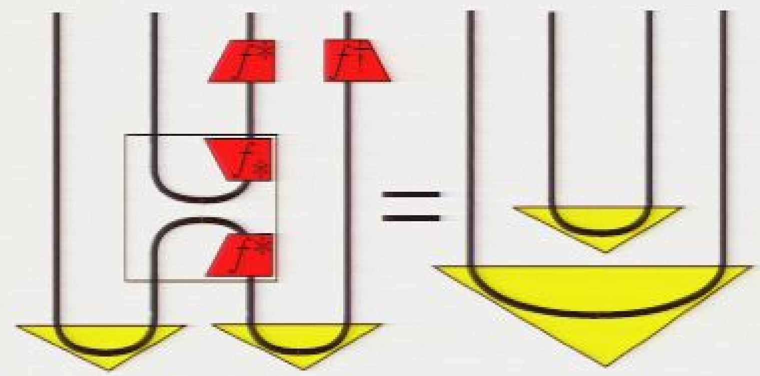
String diagrams in Quantum Protocols



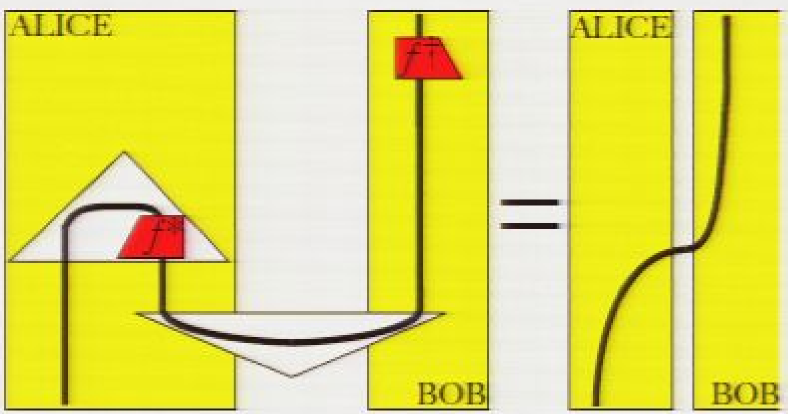


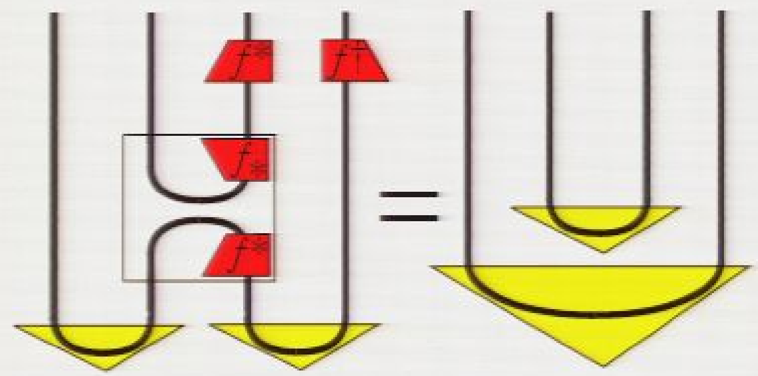
String diagrams in Quantum Protocols



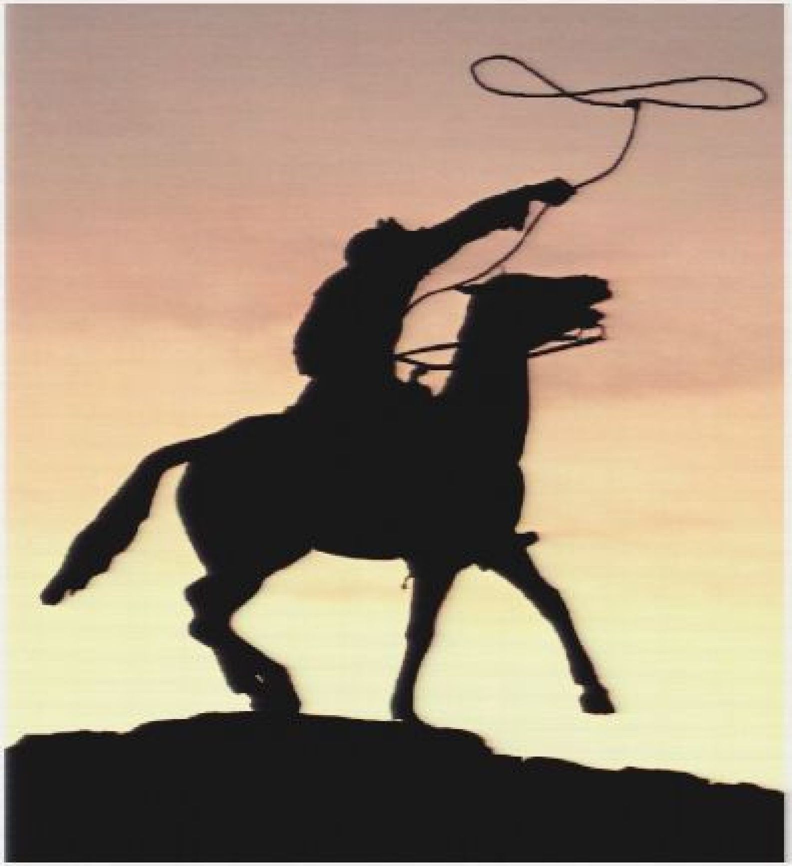


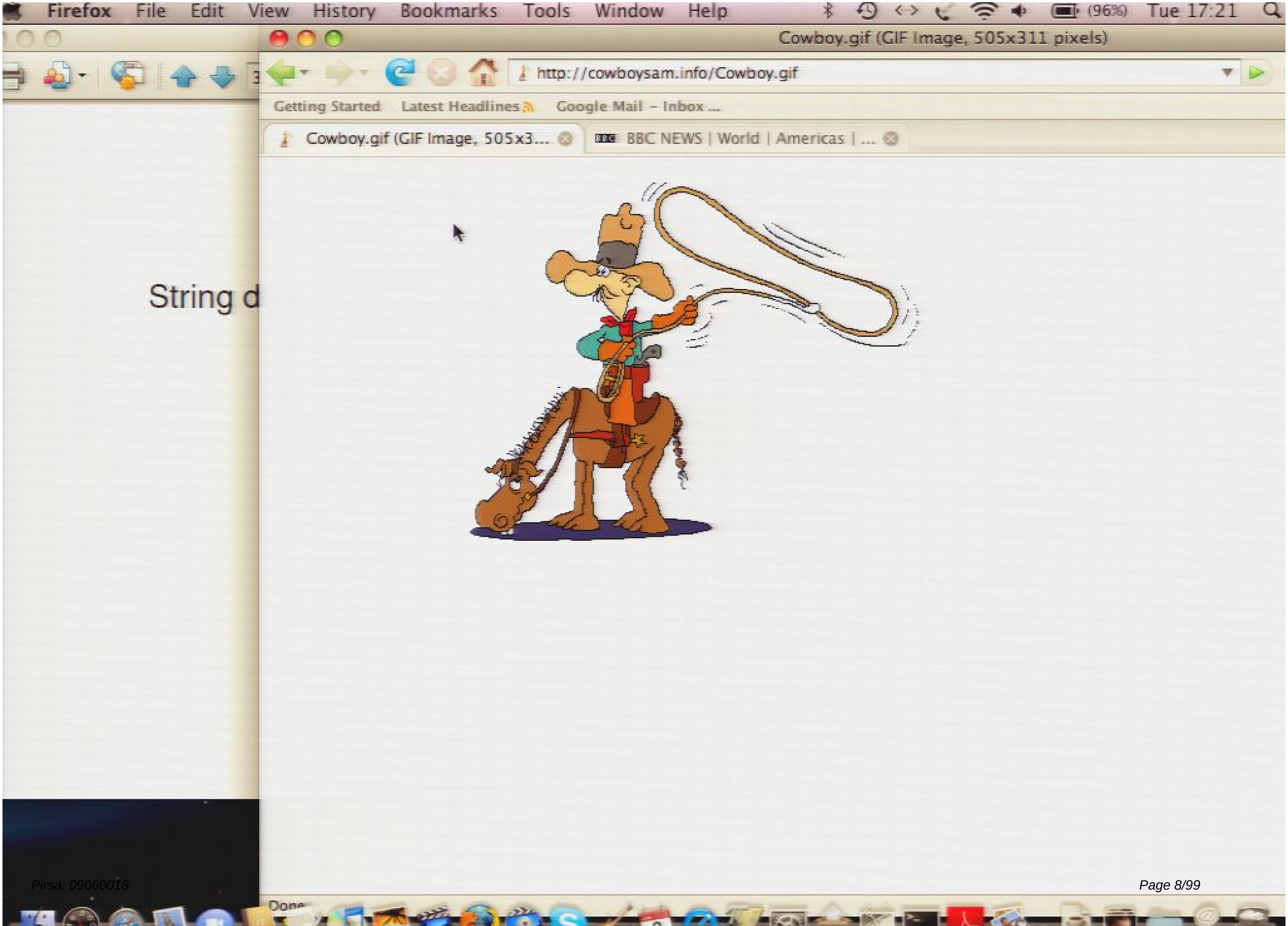
String diagrams in Quantum Protocols

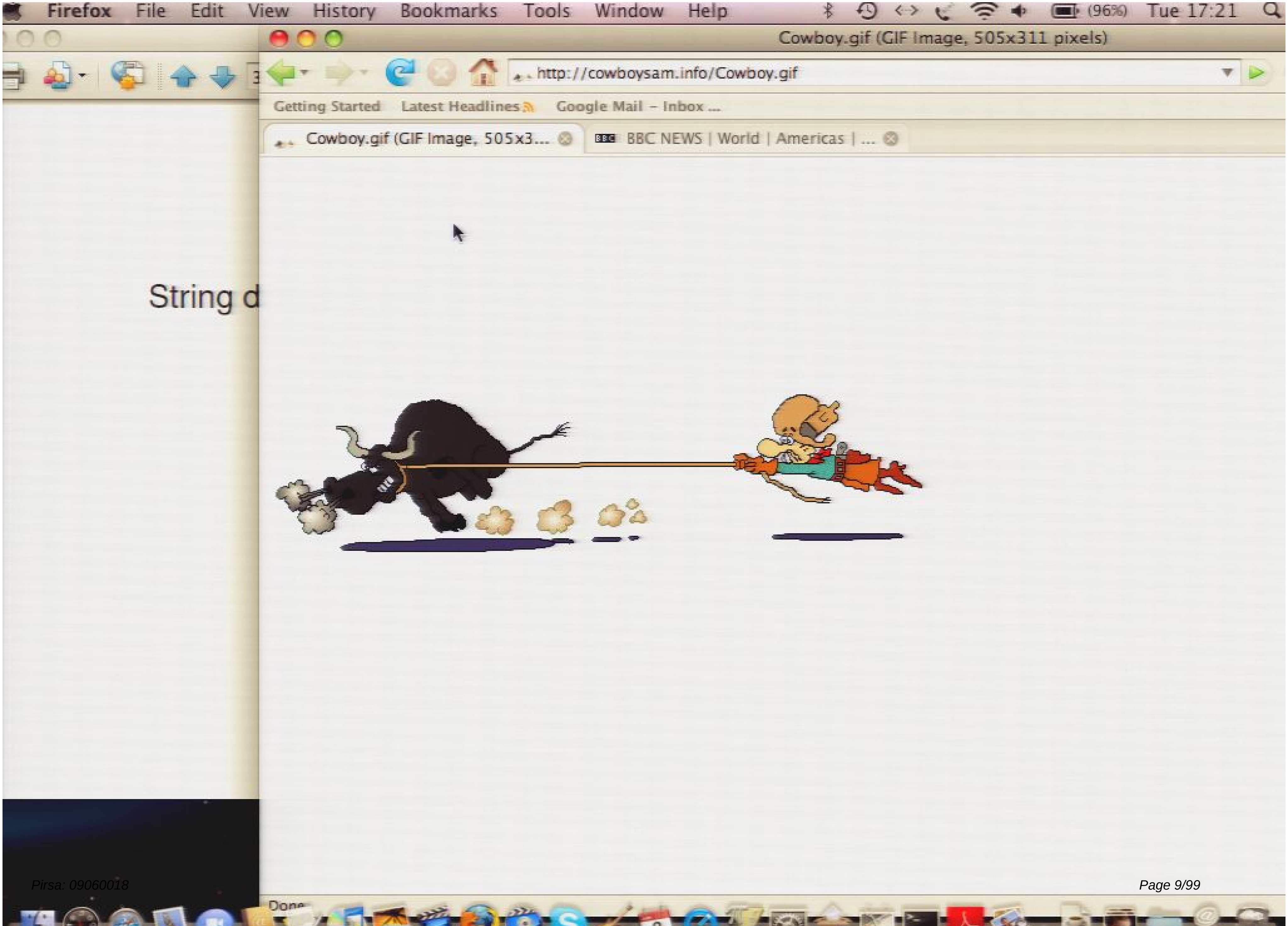




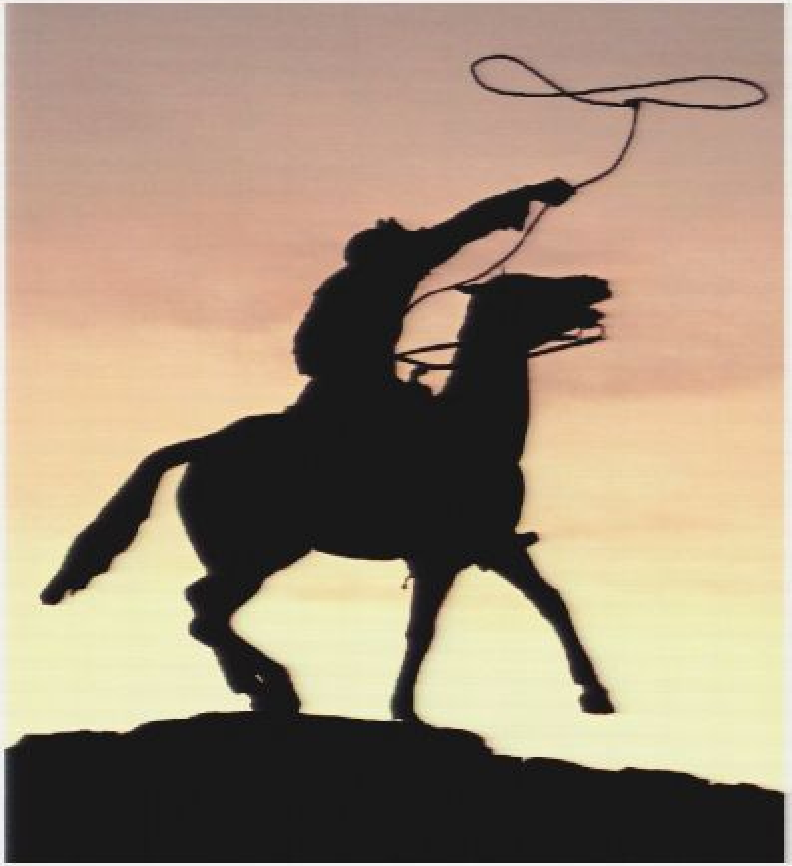
String diagrams in Western Movies




Firefox File Edit View History Bookmarks Tools Window Help
(96%) Tue 17:21
Cowboy.gif (GIF Image, 505x311 pixels)
http://cowboysam.info/Cowboy.gif
Getting Started
Latest Headlines
Google Mail – Inbox ...
Cowboy.gif (GIF Image, 505x3...
BBC NEWS | World | Americas | ...
String d
Done


Firefox File Edit View History Bookmarks Tools Window Help
(96%) Tue 17:21
Cowboy.gif (GIF Image, 505x311 pixels)
http://cowboysam.info/Cowboy.gif
Getting Started Latest Headlines Google Mail – Inbox ...
Cowboy.gif (GIF Image, 505x3...
BBC NEWS | World | Americas | ...
String d
Done

String diagrams in Western Movies



String diagrams at lunch

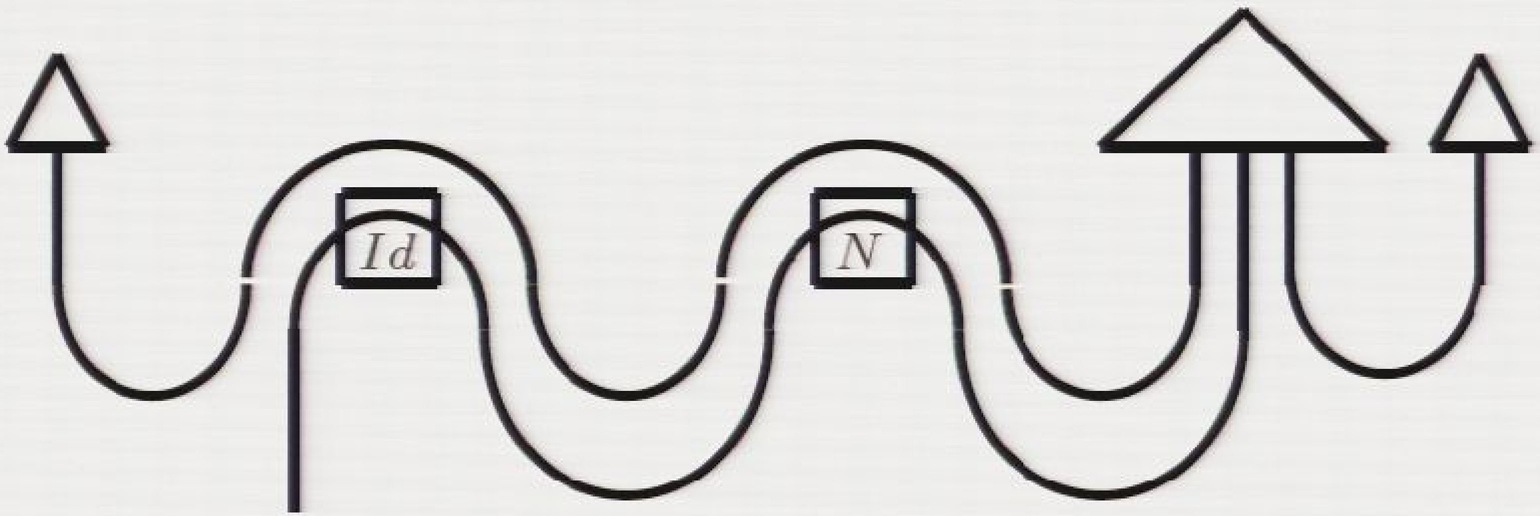
String diagrams while combing the hair

String diagrams in the garden

String diagrams in nature

String diagrams in heaven

String diagrams in natural languages



What is the vector space content of what we say?

A Categorical Approach to Distributed Meaning

Mehrnoosh Sadrzadeh
Computing Lab, University of Oxford

Joint work with: S. Clark, B. Coecke, A. Preller

Linguistics

Analyzing Natural Languages

$\begin{cases} \text{Grammar} \\ \text{Meaning} \end{cases}$

Mathematical Linguistics

Mathematical Structures for Analyzing Natural Languages

$$\begin{cases} \text{Syntax} \\ \\ \text{Semantics} \end{cases}$$

Analyzing syntax

(I) The algebraic way: type-logical (categorial) grammars

$$\begin{cases} \text{Assign types to constituents of a phrase} \\ \\ \text{Compose the types to get the type of the phrase} \end{cases}$$

(II) The Chomsky way: write the grammatical rules of a language as rewrite rules to generate the syntax of a language.

Analyzing semantics

(I) Symbolic meaning

Assigning sets to types,
use logical connectives to connect them.

(II) Distributional (vector space) meaning

Assigning vectors to words in a high-dimensional vector space,
bases are chosen according to the domain of meaning.

Analyzing semantics

(I) Symbolic meaning

- Pros:
  - Compositional,
  - Model-theoretic semantics (Montague),
  - Automated inferences.
- Cons:
  - Qualitative (true-false),
  - Says very little about lexical semantics,
  - Not very suitable for real world text.

Analyzing semantics

(II) Distributional (vector space) meaning

$$\begin{cases} \text{Cons: Non-compositional.} \\ \\ \text{Pros:} \begin{cases} \text{Quantitative,} \\ \text{All about lexical semantics.} \end{cases} \end{cases}$$

Can we develop a formalism that has the best of the two?

$$\begin{cases} \text{Pros: Compositional,} \\ \\ \text{Pros: Quantitative.} \end{cases}$$

$$\boxed{\begin{array}{c}\text{Compact Closed Categories} \\ \text{via} \\ \text{Quantum Informatique Diagrams}\end{array}}$$

Analyzing syntax

The algebraic way: type-logical (categorial) grammars

$$\begin{cases} \text{Assign types to constituents of a phrase} \\ \\ \text{Compose the types to get the type of the phrase} \end{cases}$$

Syntax Calculus. PO Monoid with residuated multiplication

$$(P, \leq, \bullet, 1, \rightarrow, \leftarrow)$$

Pregroups. PO Monoid with residuated elements

$$(P, \leq, \bullet, 1, (-)^l, (-)^r)$$

Google	bought	Microsoft.		
np	vp	np	$\leq$	?
np	$(np^r\ s\ np^l)$	np	$\leq$	s

Pregroup

$$(P, \leq, \bullet, 1, (-)^l, (-)^r)$$

A partially ordered monoid whose every element has a left and a right adjoint.

$(P, 1)$ a monoid

$$p, q \in P \implies pq \in P, \quad qp \in P, \quad p1 = 1p = p$$

$(P, 1, \leq)$ po-monoid

$$p \leq q \implies pp_1 \leq qp_1 \qquad p_1 p \leq p_1 q$$

Each element has a left & a right adjoint

$$p \in P \implies p^r \in P, \quad p^l \in P$$

$$p^l p \leq 1 \leq pp^l \qquad pp^r \leq 1 \leq p^r p$$

Some Properties of a Pregroup

Adjoint are unique and anti-tone

$$p \leq q \quad \Longrightarrow \quad q^l \leq p^l, \quad q^r \leq p^r$$

The unit is self adjoint

$$1^l = 1^r = 1$$

The multiplication is self adjoint

$$(p \bullet q)^l = q^l \bullet p^l \qquad (p \bullet q)^r = q^r \bullet p^r$$

Pregroup Grammars

Let Σ be the set of words of a natural language and $\mathcal{B}$ a POset.

Def1. 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 pregroup generated over the partial order $\mathcal{B}$, as constructed by Lambek.

Def2. A **Pregroup grammar** is a pair

$$G = \langle D, \alpha \rangle$$

of a pregroup dictionary and a set of distinguished elements $\alpha \subset \mathcal{B}$.

Def3. A string of words $w_1 \ldots w_n$ of Σ is **grammatical** if and only if

$$t_1 \cdots t_n \leq s \in \alpha$$

in $T(\mathcal{B})$, where each (w_i, t_i) is a element in D.

Example of a Pregroup Grammar

A dictionary that generates sentences "Bob likes beer." and "Bob does not like beer." has the following types

Bob	:	n	does	:	$n^r s j^l \sigma$
likes	:	$n^r s n^l$	not	:	$\sigma^r j j^l \sigma$
beer	:	n	like	:	$\sigma^r j n^l$

The basic types n, s, j stand for noun phrase, statement and infinitive; σ is an index type.

Based on these types, the above sentences are grammatical; their reductions are morphisms in $T(\mathcal{B})$.

Analyzing syntax with Pregroups

Assign types to constituents

Bob	likes	beer
n	$(n^r s\, n^l)$	n

Use axioms of the pregroups to reduce the composition of types

Bob	likes	beer.	$\Longrightarrow$	statement
n	$(n^r s\, n^l)$	n	$\leq$	s

Bob	does	not	like	beer.	$\Longrightarrow$	statement
n	$(n^r s j^l \sigma)$	$(\sigma^r j j^l \sigma)$	$(\sigma^r j n^l)$	n	$\leq$	s

Diagrammatic Analysis of Syntax in Pregroups

Bob	likes	beer.
n	$(n^r s n^l)$	n

Bob	does	not	like	beer.
n	$(n^r s j^l \sigma)$	$(\sigma^r j j^l \sigma)$	$(\sigma^r j n^l)$	n

Pregroups and Natural Languages

English, French, Italian, German, Latin, Polish, Turkish, Japanese, Arabic, Persian, Hebrew, Mandarine, ...

گویند بهشت با حور خوش است

من می گو یم که آب انگور خوش است

این نقد بگیر و دست از آن نسیه بدار

کاواز دهل شنیدن از دور خوش است

Diagrammatic Analysis of Syntax in Pregroups

Bob	likes	beer.
n	$(n^r s\, n^l)$	n

Bob	does	not	like	beer.
n	$(n^r s j^l \sigma)$	$(\sigma^r j j^l \sigma)$	$(\sigma^r j n^l)$	n

Pregroups and Natural Languages

English, French, Italian, German, Latin, Polish, Turkish, Japanese, Arabic, Persian, Hebrew, Mandarine, ...

گویند بهشت با حور خوش است

من می گو یم که آب انگور خوش است

این نقد بگیر و دست از آن نسیه بدار

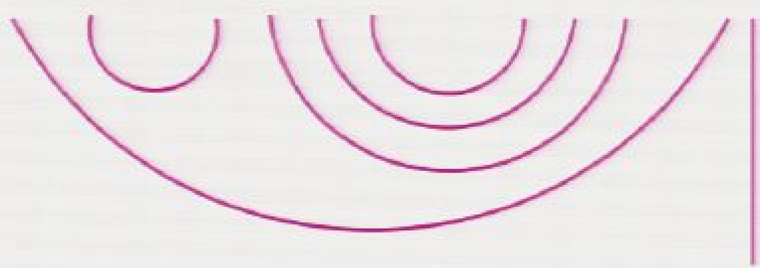
کاواز دهل شنیدن از دور خوش است

Comparing Syntactic Structure of Sentences in Pregroups

English and French

Arabic and Hebrew

Persian and Hindi



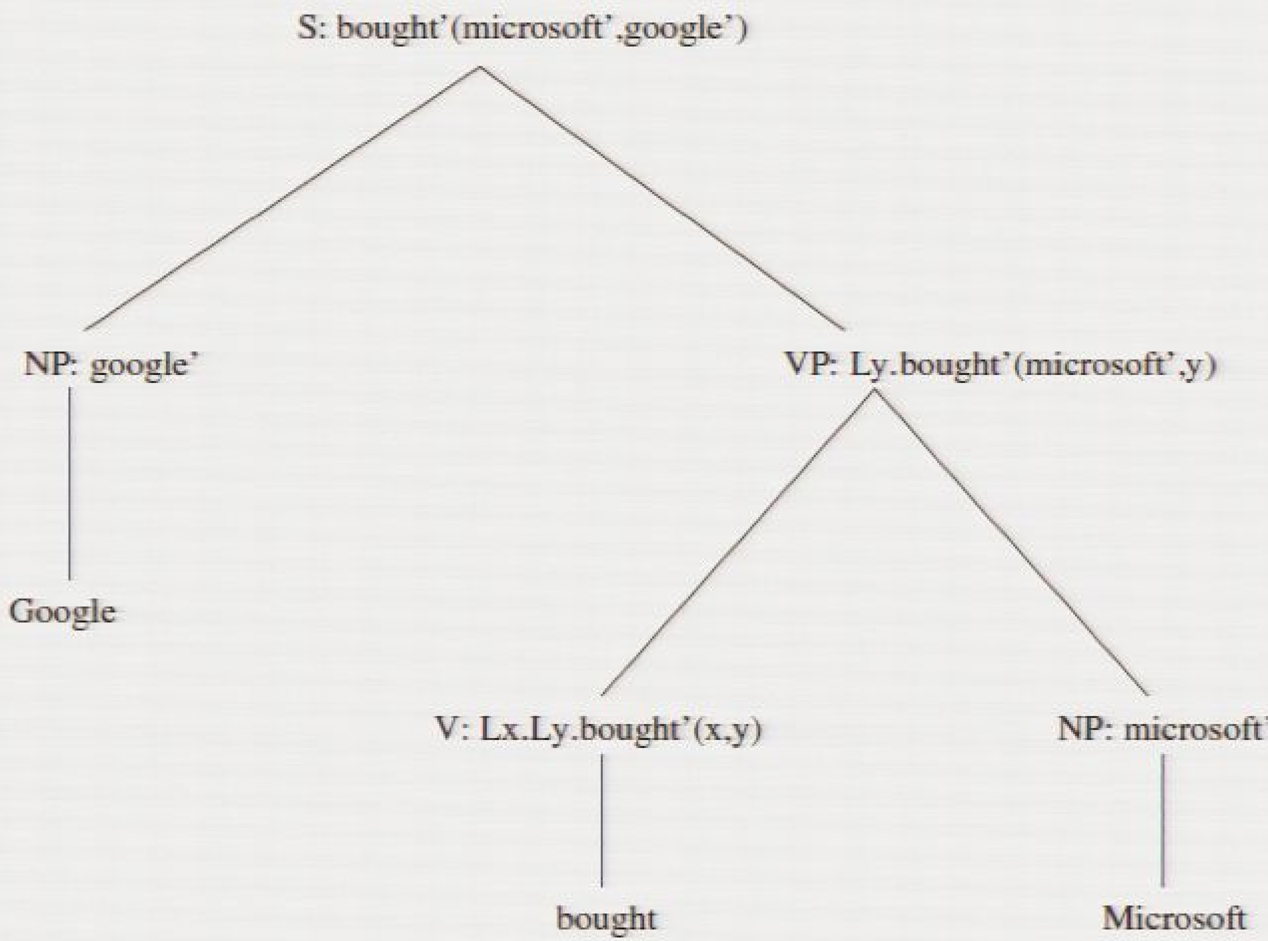
Symbolic Model of Meaning

Meaning of a sentence is a function of meaning of its constituents.

This function is obtained by composing the meaning functions of the words within the sentence.

Some makes sense: verbs are relations

Some do not make sense: nouns are sets


S: bought'(microsoft',google')
NP: google'
VP: Ly.bought'(microsoft',y)
Google
V: Lx.Ly.bought'(x,y)
NP: microsoft'
bought
Microsoft

Distributional (Vector-Based) Model of Meaning

Firth: ”You shall know a word by the company it keeps”.

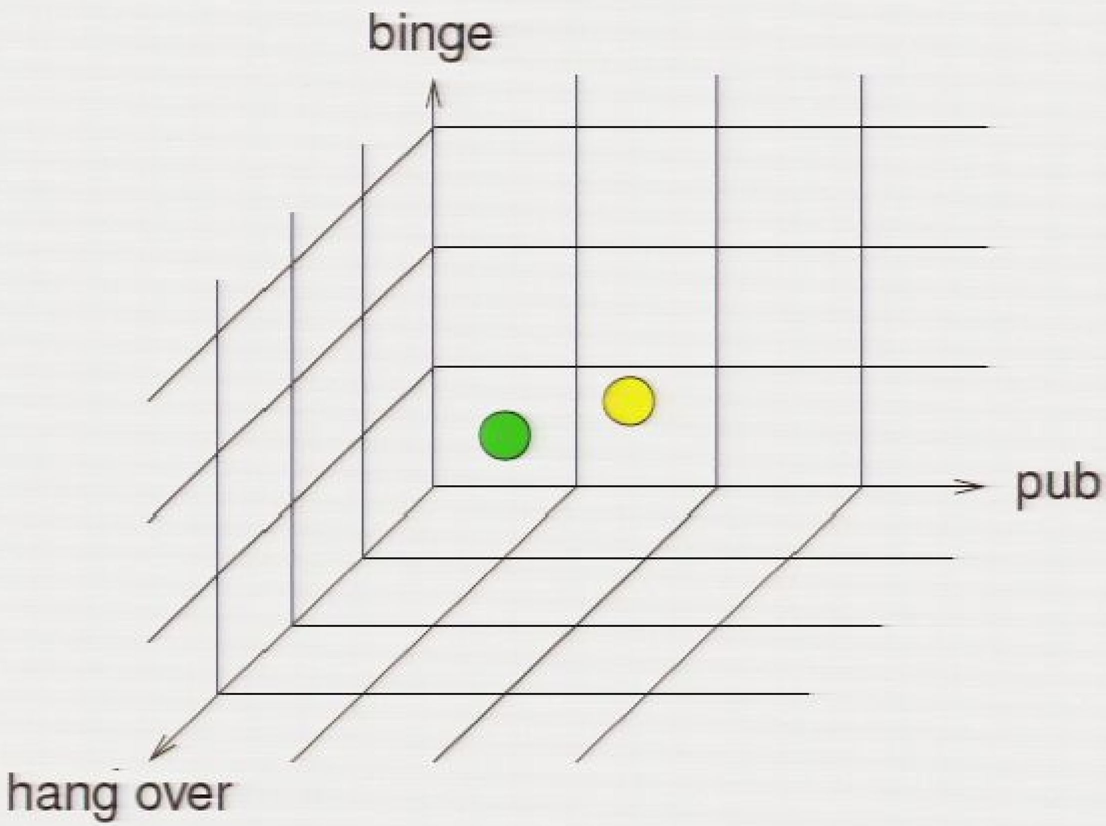
Intuition: the meanings of *beer* and *whisky* are similar (in some way) because they both get you drunk, are served at the pub, have alcohol, damage your liver, cause a hang over if binged on, etc.

These facts are reflected in text: *beer* and *whisky* both appear close to the words *drunk, pub, alcohol, liver, hang over, binge*.

In the same way, there is a similarity between the words *cat* and *dog*, also between *ship* and *boat*, etc.

In this approach meaning vectors live in a high-dimensional ”semantic space”, where **context** is often just an n-word window.

Meanings as Context Vectors



Thesaurus Construction

Curran (2003): *From Distributional to Semantic Similarity*

Created context vectors from 2 billion words of text

Compare context vectors to find pairs of synonyms

Example Thesaurus

introduction: launch, implementation, advent, addition, adoption, arrival, absence, inclusion, creation

evaluation: assessment, examination, appraisal, review, audit, analysis, consultation, monitoring, testing, verification

methods: technique, procedure, means, approach, strategy, tool, concept, practice, formula, tactic

Question:

How to bring the compositionality of the algebraic typing to the lexical ability of distributed meaning?

$$
\begin{tikzcd}
 & \textit{a mathematical structure} \arrow[dl, "\textit{data}"'] \arrow[dr, "\textit{data}"] & \\
\textit{vector space} \arrow[dr, "\textit{meaning}"'] & & \textit{pregroup} \arrow[dl, "\textit{type logic}"] \\
 & \textit{language} &
\end{tikzcd}
$$

Example Thesaurus

introduction: launch, implementation, advent, addition, adoption, arrival, absence, inclusion, creation

evaluation: assessment, examination, appraisal, review, audit, analysis, consultation, monitoring, testing, verification

methods: technique, procedure, means, approach, strategy, tool, concept, practice, formula, tactic

Question:

How to bring the compositionality of the algebraic typing to the lexical ability of distributed meaning?

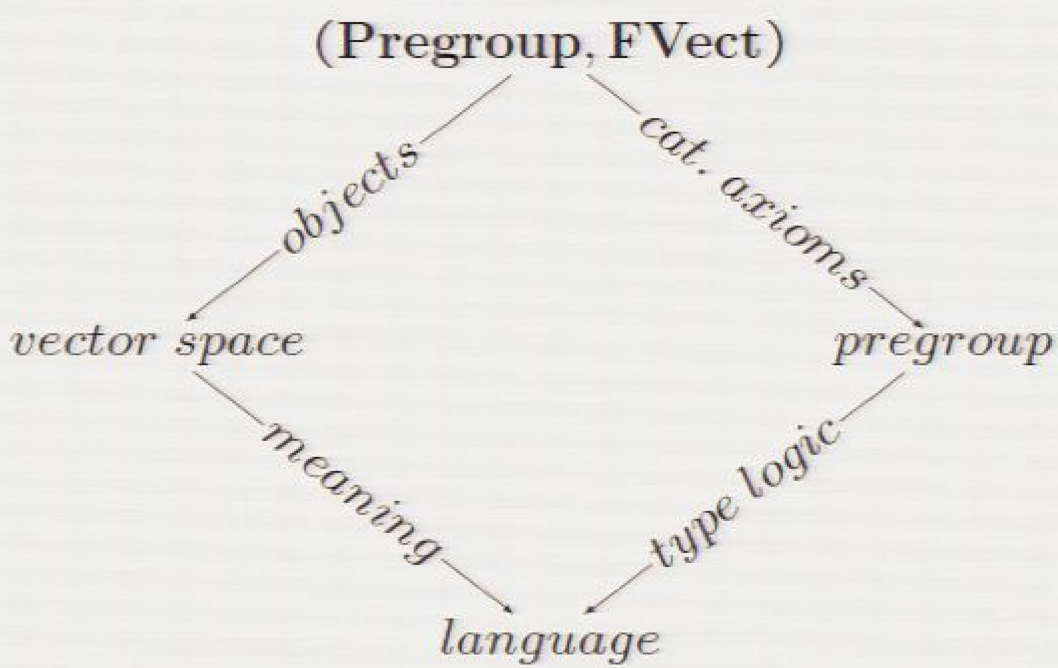
$$
\begin{tikzcd}
 & \textit{a mathematical structure} \arrow[dl, "\textit{data}"'] \arrow[dr, "\textit{data}"] & \\
\textit{vector space} \arrow[dr, "\textit{meaning}"'] & & \textit{pregroup} \arrow[dl, "\textit{type logic}"] \\
 & \textit{language} &
\end{tikzcd}
$$

Finite dimensional vector spaces are compact closed categories.

Pregroups are compact closed categories.

Answer:

Compact Closed Categories

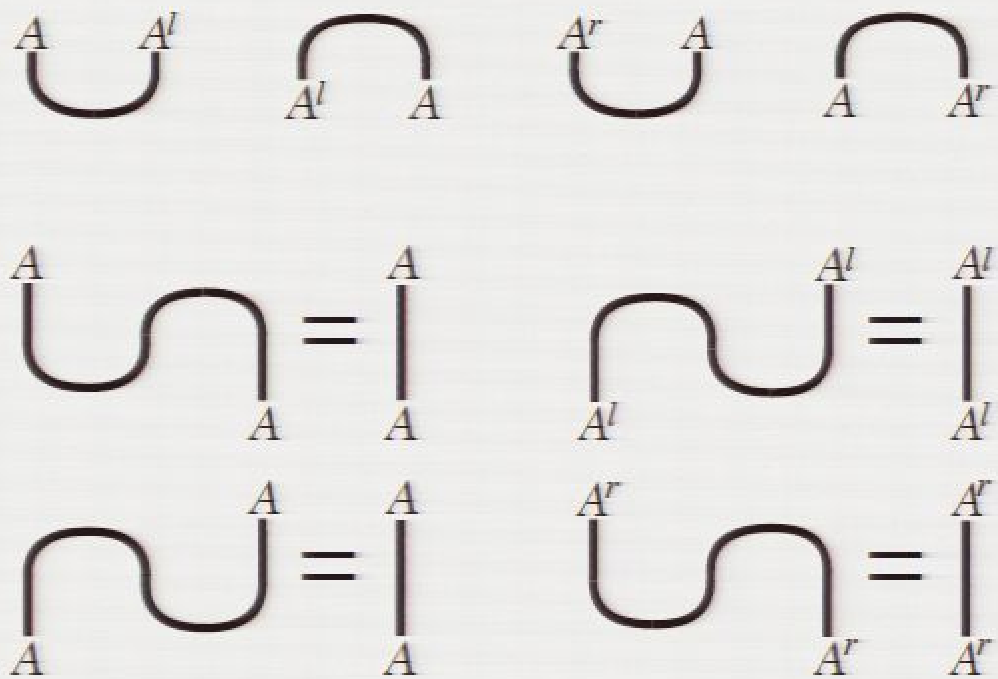


Compact Closed Category

A monoidal closed category, where each object has a left & a right adjoint satisfying yanking axioms.

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

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



A Pregroups is Compact Closed.

1- Elements of pregroup are objects $p, q \in P$
2- Partial order is the morphism $p \to q$ iff $p \leq q$
3- Tensor is monoid multiplication $p \otimes q$ iff pq, unit is 1
4- Adjoints are adjoints.
5- Epsilon maps are

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

6- Eta maps are

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

Bob	likes	beer	$\Longrightarrow$	statement
n	$(n^r s n^l)$	n	$\leq$	s

The reduction of types becomes a morphism in the category

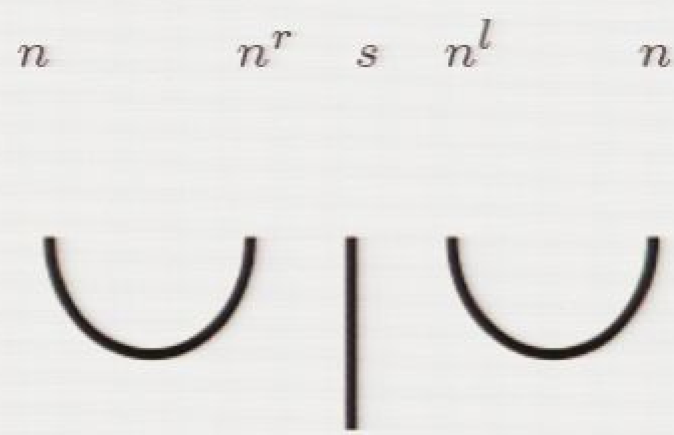
$$\epsilon_n^r \otimes 1_s \otimes \epsilon_n^l : n \otimes n^r \otimes s \otimes n^l \otimes n \to s$$

Reductions become Morphisms

The reduction morphism of ”Bob likes beer” is

$$\epsilon_n^r \otimes 1_s \otimes \epsilon_n^l$$

depicted as



Reduction become Morphisms

The reduction morphism of "Bob does not like beer" is

$$\left(1_s \otimes \epsilon_j^l \otimes \epsilon_j^l\right) \circ \left(\epsilon_n^r \otimes 1_{sj^l} \otimes \epsilon_\sigma^r \otimes 1_{jj^l} \otimes \epsilon_\sigma^r \otimes 1_j \otimes \epsilon_n^l\right)$$

It is depicted as follows

$$n \qquad n^r s j^l \sigma \qquad \sigma^r j j^l \sigma \qquad \sigma^r j n^l \qquad n$$

FVector Spaces are Compact Closed.

1- Vector spaces are objects V, W
2- Linear maps are morphisms $f: V \to W$
3- Tensor is tensor $V \otimes W$, unit is $\mathbb{R}$.
4- Adjoints are identity $V^l = V = V^r$
5- Given a base $\{r_i\}_i$, epsilon maps are inner products

$$\epsilon^l = \epsilon^r : V \otimes V \to \mathbb{R}$$

$$\sum_{ij} c_{ij} \psi_i \otimes \phi_j \mapsto \sum_{ij} c_{ij} \langle \psi_i | \phi_j \rangle \, .$$

6- Eta maps create Bell states

$$\eta^l = \eta^r : \mathbb{R} \to V \otimes V$$

$$1 \mapsto \sum_i e_i \otimes e_i$$

These are maximally entangled states that allow for the non-loal co-relations of Quantum Mechanics.

Semantics for Pregroups: Quantizing Functor

$$\text{Pregroups} \xrightarrow{[\![\ ?\ ]\!]} \text{Vector Spaces}$$

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 $T(D)$.

Example of a Quantizing Functor

$$[\![(\text{Bob}, n)]\!] = V$$

$$[\![(\text{beer}, n)]\!] = W$$

$$[\![(\text{likes}, n^r \otimes s \otimes n^l)]\!] = V^* \otimes S \otimes W^*$$

$$[\![(\text{like}, \sigma^r \otimes j \otimes n^l)]\!] = V^* \otimes J \otimes W^*$$

$$[\![(\text{does}, n^r \otimes s \otimes j^l \otimes \sigma)]\!] = V^* \otimes S \otimes J^* \otimes V$$

$$[\![(\text{not}, \sigma^r \otimes j \otimes j^l \otimes \sigma)]\!] = V^* \otimes J \otimes J^* \otimes V$$

Compositional Meaning

The meaning vector of a string of words from a language Σ

$$w_1 \ldots w_n$$

with type assignments $(w_i, t_i) \in D$ and a syntactic reduction map

$$t_1 \cdots t_n \xrightarrow{f} s$$

is

$$\overrightarrow{w_1 \ldots w_n} := \left\langle [\![f]\!] \circ \tilde{\eta} \,\middle|\, \overrightarrow{w_1} \otimes \ldots \otimes \overrightarrow{w_n} \right\rangle$$

where $\tilde{\eta}$ is a series of η maps and each w_i lives in $[\![(w_i, t_i)]\!]$.

Example: Positive Transitive Sentence

Meaning of ”Bob likes beer” is obtained by applying the semantic map of its syntactic reduction to the tensor product of vectors of the words therein:

$$\overrightarrow{Bob\ \ likes\ \ beer} \quad = \quad \Big(\langle\epsilon_V| \otimes \mathbf{1}_S \otimes \langle\epsilon_W|\Big) \Big|\overrightarrow{Bob} \otimes \overrightarrow{likes} \otimes \overrightarrow{beer}\Big\rangle$$

$\Big\langle$  $\Big|$ $\overrightarrow{Bob} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{beer}\Big\rangle$

Concrete Calculations

Given

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

We obtain

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

$$\sum_k \left(\sum_{ij} C_{ik} \langle \overrightarrow{Bob} | \overrightarrow{v}_i \rangle \langle \overrightarrow{w}_j | \overrightarrow{beer} \rangle \right) \overrightarrow{s}_k \,.$$

Concrete Calculations for Boolean Meaning

V is spanned by all men $\{\overrightarrow{m_i}\}_i$

W is spanned by all drinks $\{\overrightarrow{d_j}\}_j$

S is spanned by two vectors $|1\rangle$ and $|0\rangle$, denoting *true* and *false*.

The verb "likes" becomes the following superposition

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

Concrete Calculations for Boolean Meaning

To get a truth-theroetic meaning we can set

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

Assume Bob is m_3 and beer is d_4.

The meaning of our sentence becomes

$$\sum_{ij} \left\langle \overrightarrow{m}_3 \mid \overrightarrow{m_i} \right\rangle \otimes \overrightarrow{s}_{ij} \otimes \left\langle \overrightarrow{d}_j \mid \overrightarrow{d_4} \right\rangle = \sum_{ij} \delta_{3i} \overrightarrow{s}_{ij} \delta_{j4} = \overrightarrow{s}_{34}$$

This is *true* if ”Bob likes beer” and *false* otherwise.

Weighted Meaning

Assume 'like' has degrees of "love" and "hate", e.g.

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

$$\overrightarrow{loves} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{loves_{ij}} \otimes \overrightarrow{d_j}, \qquad \overrightarrow{hates} = \sum_{ij} \overrightarrow{m_i} \otimes \overrightarrow{hates_{ij}} \otimes \overrightarrow{d_j}$$

where $\overrightarrow{loves_{ij}}$ and $\overrightarrow{hates_{ij}}$ have Boolean meanings.

Assume S is spanned by "love" and "hate".

Now the meaning of sentence "Bob likes beer" is a vector in the vector space whose basis are "love" and "hate".

In particular, it is true whenever

$$\overrightarrow{Bob\ \ likes\ \ beer} \quad = \quad \begin{pmatrix} 3/4 \\ 1/4 \end{pmatrix}$$

Example: Negative Transitive Sentence

The meaning map of the sentence ”Bob does not like beer” is

$$\overrightarrow{Bob\ \ does\ \ not\ \ like\ \ beer} \ \ =$$

$$\left\langle f \circ \tilde{\eta} \,\middle|\, \overrightarrow{Bob} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{beer} \right\rangle$$

Meaning Morphisms for Negative Sentence

The process of computing meaning will have two steps:

(I) Eta maps to create extra space for temporary substitutions $\tilde{\eta}$

$$(1_{V \otimes V^*} \otimes \eta_{S=J} \otimes 1_{V \otimes V^*} \otimes \eta_J \otimes 1_V \otimes 1_{V^* \otimes J \otimes W^*} \otimes 1_W) \circ (1_V \otimes \eta_V \otimes \eta_V \otimes 1_{V^* \otimes J \otimes W^*} \otimes 1_W)$$

(II) Epsilon maps for substitution f

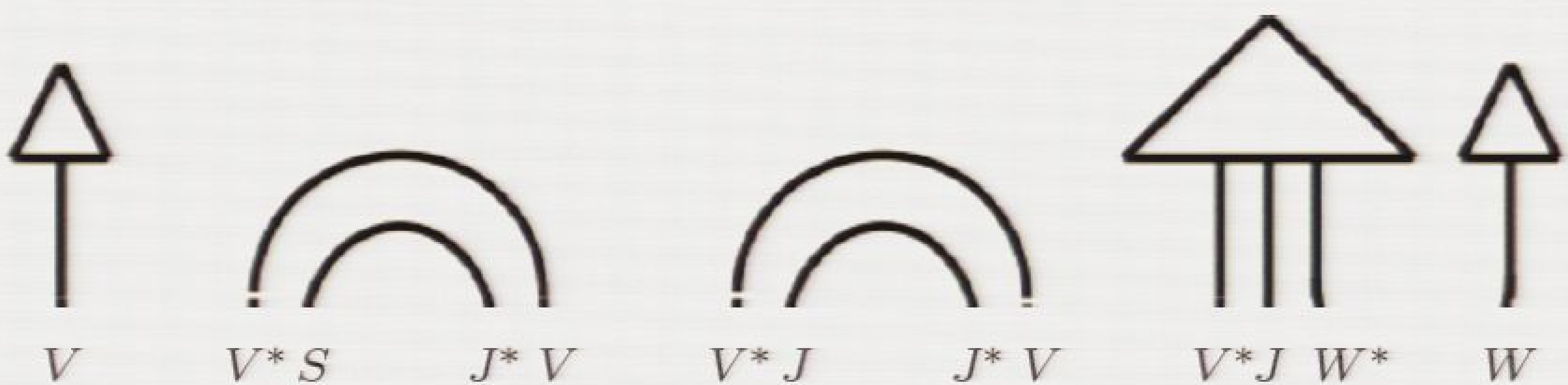
$$(1_S \otimes \epsilon_J \otimes \epsilon_J) \circ (\epsilon_V \otimes 1_S \otimes 1_{J*} \otimes \epsilon_V \otimes 1_J \otimes 1_{J*} \otimes \epsilon_V \otimes 1_J \otimes \epsilon_W)$$

Meaning Morphisms for Negative Sentence

$$(1_{V\otimes V^*}\otimes\eta_{S=J}\otimes 1_{V\otimes V^*}\otimes\eta_J\otimes 1_V\otimes 1_{V^*\otimes J\otimes W^*}\otimes 1_W)\circ(1_V\otimes\eta_V\otimes\eta_V\otimes 1_{V^*\otimes J\otimes W^*}\otimes 1_W)$$

Assume $S = J$ and create 4 Bell states, i.e. functions

$$V^*\otimes V \equiv V\to V \qquad J^*\otimes S\equiv J\to S \qquad J^*\otimes J\equiv J\to J$$



Meaning Morphisms for Negative Sentence

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

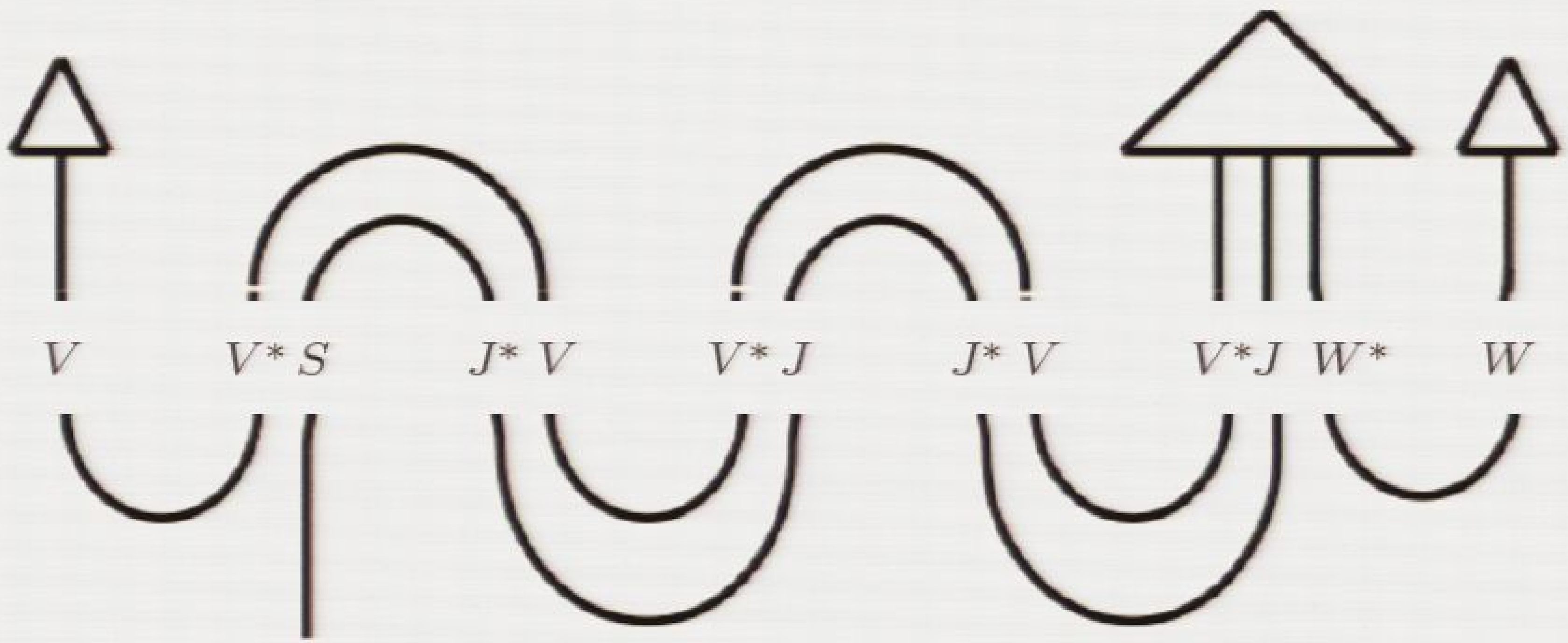
Substitution

$$\epsilon_V^*: V \otimes V \to 1 \qquad \epsilon_J: J^* \otimes J \to 1$$

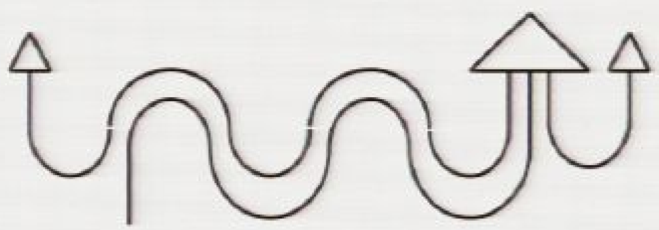
Meaning Morphisms for Negative Sentence

The full map of the meaning is obtained by the composition

$$f \circ \tilde{\eta}$$



$$\overrightarrow{Bob\ does\ not\ like\ beer} \quad =$$

$\langle$  $| \ \overrightarrow{Bob} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \overrightarrow{beer} \rangle$

Concrete Calculations for Boolean Meaning

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

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

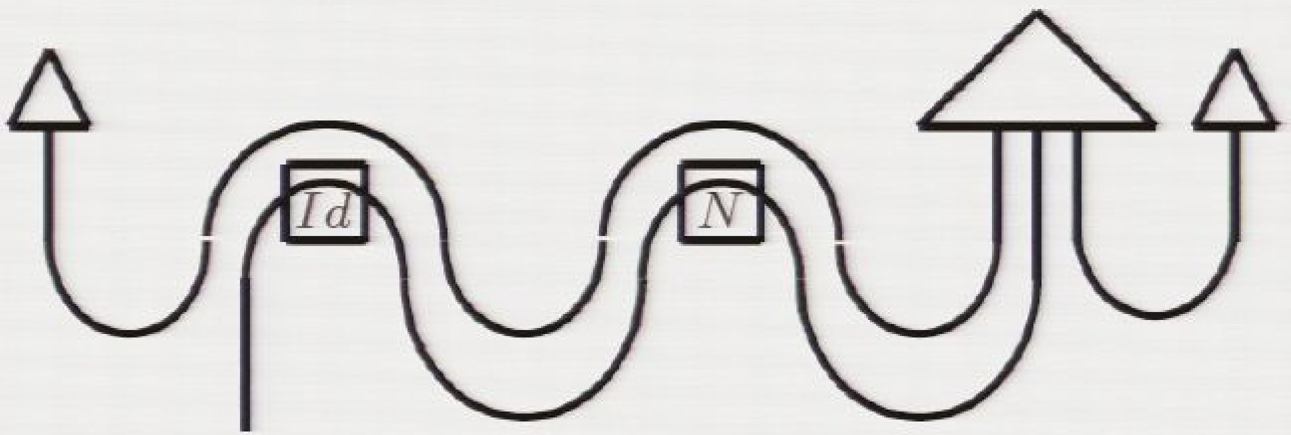
$$\overrightarrow{does} = \sum_l \overrightarrow{m}_l \otimes (|11\rangle + |00\rangle) \otimes \overrightarrow{m}_l \in V^* \otimes S \otimes J^* \otimes V$$

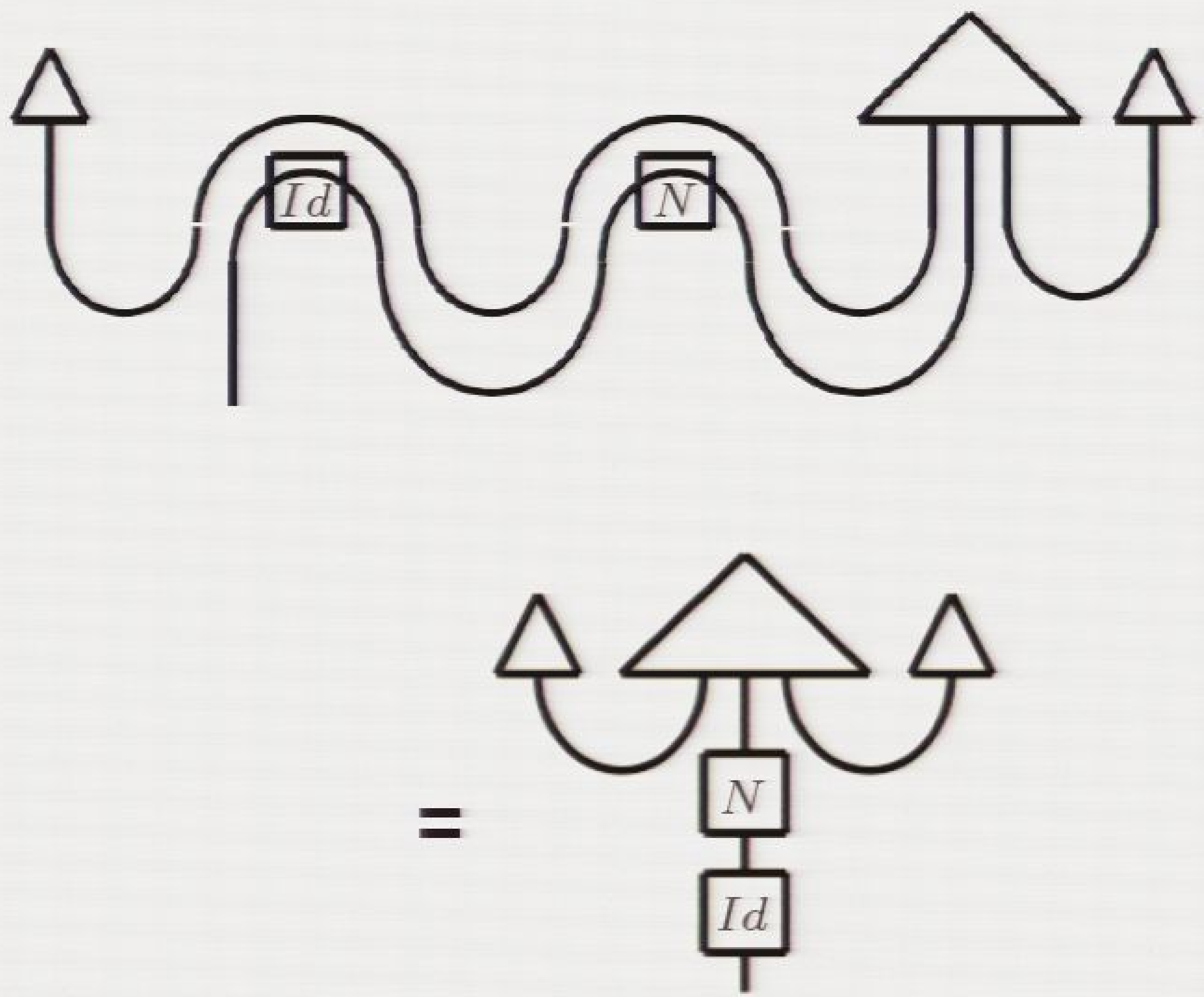
Assuming that Bob is m_3 and beer is d_4 and abbreviating

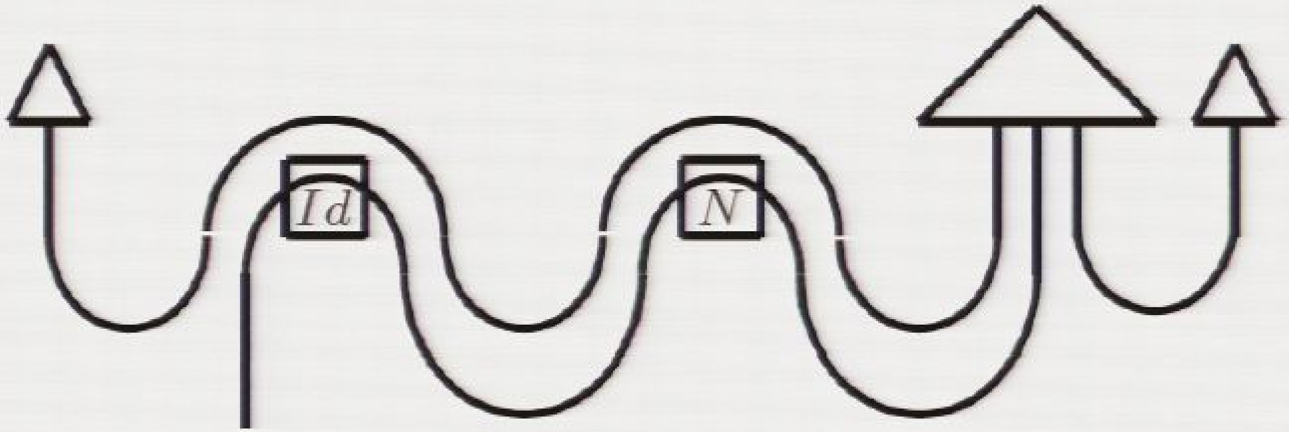
$$|10\rangle + |01\rangle \quad \text{to} \quad \overline{not}$$

$$|00\rangle + |11\rangle \quad \text{to} \quad \overline{does}$$

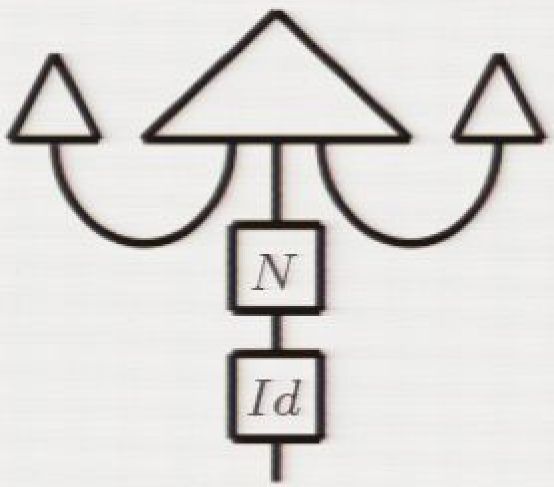
Having fixed matrices for "does", "not" the meaning map becomes:


Id
N


Id
N
=
N
Id



=



$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \mu_{34}$$

$$(f \circ g)\Big(\overrightarrow{m_3} \otimes \Big(\sum_l \overrightarrow{m}_l \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \overrightarrow{d}_j\Big) \otimes \overrightarrow{d}_4\Big) =$$

$$\Big(\sum_l \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_l\rangle \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \langle \overrightarrow{d}_j \mid \overrightarrow{d}_4\rangle\Big)$$

$$\Big(\sum_l \delta_{3l} \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \delta_{j4}\Big)$$

$$\overline{does} \otimes \overrightarrow{m}_3 \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \Big(\sum_k \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_k\rangle \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \Big(\sum_k \delta_{3k} \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{m}_3 \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

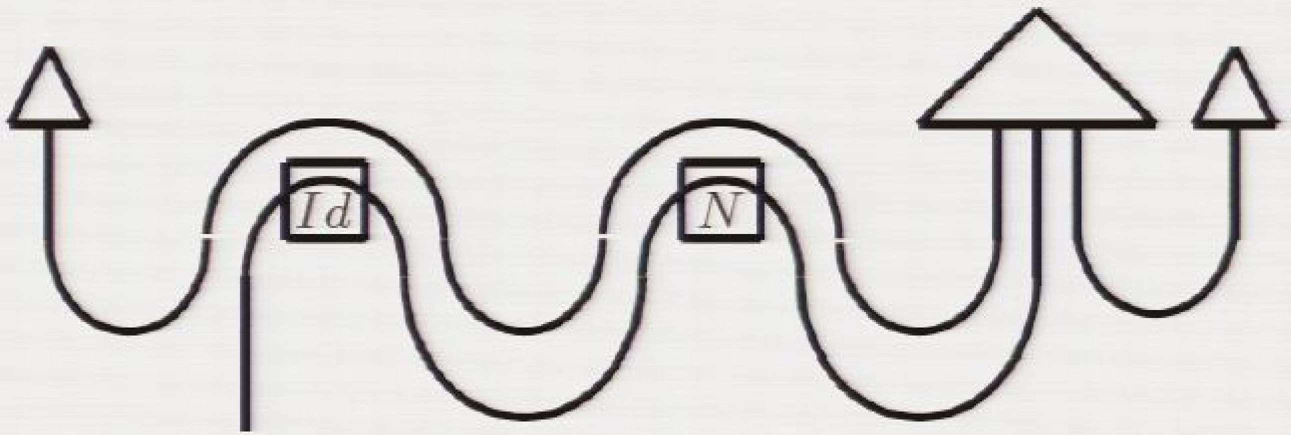
$$\overline{does} \otimes \overline{not} \otimes \Big(\sum_i \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_i\rangle \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \overline{not} \otimes \Big(\sum_i \delta_{3i} \otimes \overrightarrow{\mu}_{i4}\Big)$$

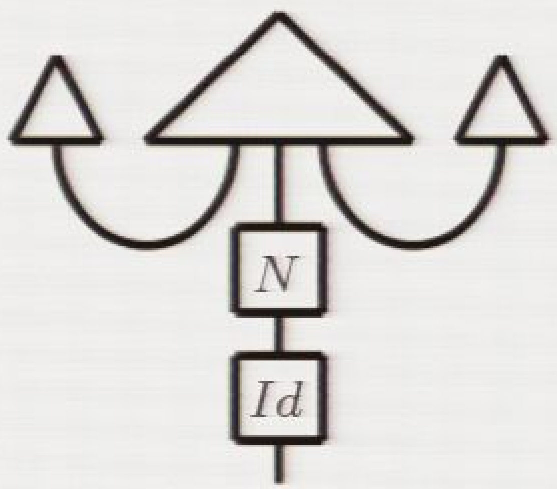
$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{\mu}_{34}$$

$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{\mu}_{34}$$

$$
\begin{aligned}
&= (|00\rangle + |11\rangle) \otimes (|10\rangle + |01\rangle) \otimes \overrightarrow{\mu}_{34} \\
&= |0\,\underbrace{01}\,\underbrace{0\overrightarrow{\mu}_{34}}\rangle + |0\,\underbrace{00}\,\underbrace{1\overrightarrow{\mu}_{34}}\rangle + |1\,\underbrace{11}\,\underbrace{0\overrightarrow{\mu}_{34}}\rangle + |1\,\underbrace{10}\,\underbrace{1\overrightarrow{\mu}_{34}}\rangle \\
&= |0\,\underbrace{1\overrightarrow{\mu}_{34}}\rangle + |1\,\underbrace{0\overrightarrow{\mu}_{34}}\rangle \\
&= \begin{cases} |0\,\underbrace{11}\rangle + |1\,\underbrace{01}\rangle & \overrightarrow{\mu}_{34} = 1 \\ |0\,\underbrace{10}\rangle + |1\,\underbrace{00}\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases} \\
&= \begin{cases} |0\rangle & \overrightarrow{\mu}_{34} = 1 \\ |1\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases}
\end{aligned}
$$



=



$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \mu_{34}$$

$$(f \circ g)\Big(\overline{m_3} \otimes \Big(\sum_l \overrightarrow{m}_l \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \overrightarrow{d}_j\Big) \otimes \overrightarrow{d}_4\Big) =$$

$$\Big(\sum_l \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_l \rangle \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \langle \overrightarrow{d}_j \mid \overrightarrow{d}_4 \rangle\Big)$$

$$\Big(\sum_l \delta_{3l} \otimes \overline{does} \otimes \overrightarrow{m}_l\Big) \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{ij} \otimes \delta_{j4}\Big)$$

$$\overline{does} \otimes \overrightarrow{m}_3 \otimes \Big(\sum_k \overrightarrow{m}_k \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \Big(\sum_k \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_k \rangle \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \Big(\sum_k \delta_{3k} \otimes \overline{not} \otimes \overrightarrow{m}_k\Big) \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{m}_3 \otimes \Big(\sum_i \overrightarrow{m}_i \otimes \overrightarrow{\mu}_{i4}\Big)$$

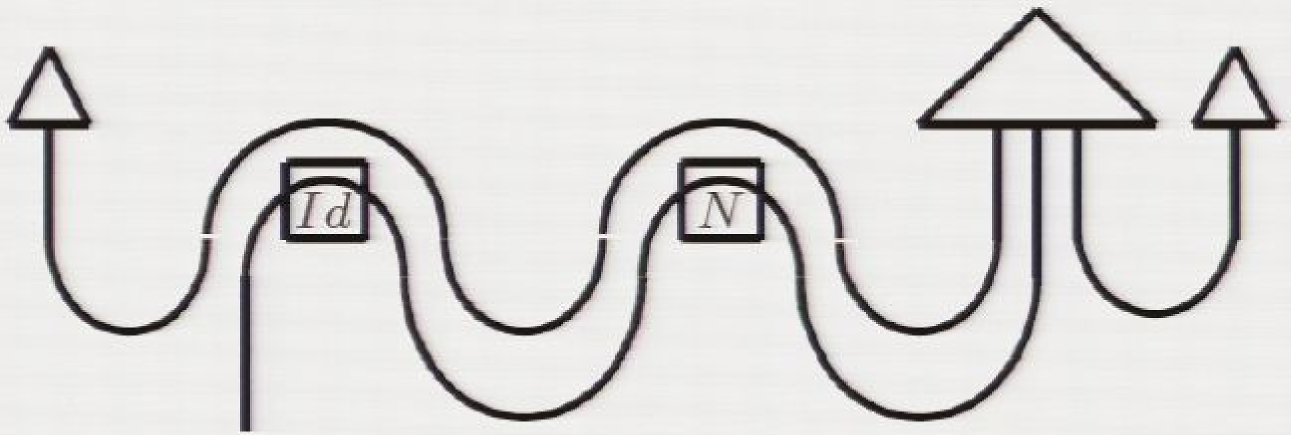
$$\overline{does} \otimes \overline{not} \otimes \Big(\sum_i \langle \overrightarrow{m}_3 \mid \overrightarrow{m}_i \rangle \otimes \overrightarrow{\mu}_{i4}\Big)$$

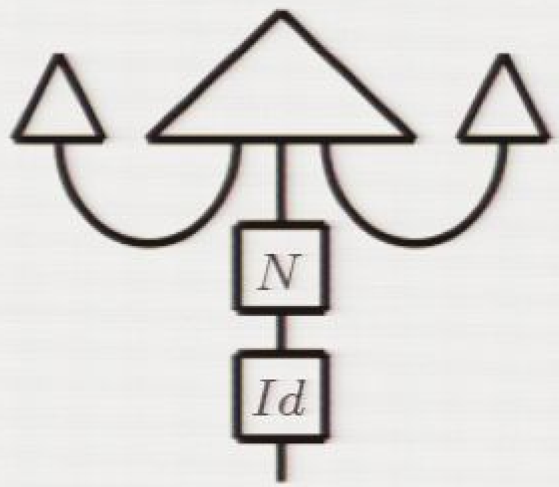
$$\overline{does} \otimes \overline{not} \otimes \Big(\sum_i \delta_{3i} \otimes \overrightarrow{\mu}_{i4}\Big)$$

$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{\mu}_{34}$$

$$\overline{does} \otimes \overline{not} \otimes \overrightarrow{\mu}_{34}$$

$$
\begin{aligned}
&= (|00\rangle + |11\rangle) \otimes (|10\rangle + |01\rangle) \otimes \overrightarrow{\mu}_{34} \\
&= |0\underbrace{01}\underbrace{0\overrightarrow{\mu}_{34}}\rangle + |0\underbrace{00}\underbrace{1\overrightarrow{\mu}_{34}}\rangle + |1\underbrace{11}\underbrace{0\overrightarrow{\mu}_{34}}\rangle + |1\underbrace{10}\underbrace{1\overrightarrow{\mu}_{34}}\rangle \\
&= |0\underbrace{1\overrightarrow{\mu}_{34}}\rangle + |1\underbrace{0\overrightarrow{\mu}_{34}}\rangle \\
&= \begin{cases} |0\underbrace{11}\rangle + |1\underbrace{01}\rangle & \overrightarrow{\mu}_{34} = 1 \\ |0\underbrace{10}\rangle + |1\underbrace{00}\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases} \\
&= \begin{cases} |0\rangle & \overrightarrow{\mu}_{34} = 1 \\ |1\rangle & \overrightarrow{\mu}_{34} = 0 \end{cases}
\end{aligned}
$$



= 

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \mu_{34}$$

Concrete Calculations for Weighted Meaning

If we define like such that it has degrees of love and hate

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

Then these degrees propagate to the negative case and the meaning of "Bob does not like beer" is obtainable by applying the Bell states of does and not to μ_{34}, that is

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3/4 \\ 1/4 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1/4 \\ 3/4 \end{pmatrix}$$

$$= \begin{pmatrix} 1/4 \\ 3/4 \end{pmatrix}$$

Concrete Calculations for Weighted Meaning

If we define like such that it has degrees of love and hate

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

Then these degrees propagate to the negative case and the meaning of "Bob does not like beer" is obtainable by applying the Bell states of does and not to μ_{34}, that is

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 3/4 \\ 1/4 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1/4 \\ 3/4 \end{pmatrix}$$

$$= \begin{pmatrix} 1/4 \\ 3/4 \end{pmatrix}$$

A Similarity Measure for Meaning of Sentences

Similarity

$$\Big\langle [\![\alpha]\!] \mid [\![\beta]\!] \Big\rangle = p$$

Examples

$$\Big\langle [\![\overrightarrow{Bob\ \ loves\ \ beer}]\!] \mid [\![\overrightarrow{Bob\ \ hates\ \ beer}]\!] \Big\rangle \ = \ 0$$

$$\Big\langle [\![\overrightarrow{Bob\ \ lovesbeer}]\!] \mid [\![\overrightarrow{Bob\ \ likes\ \ beer}]\!] \Big\rangle \ = \ \frac{3}{4}$$

$$\Big\langle [\![\overrightarrow{Bob\ \ does\ \ not\ \ like\ \ beer}]\!] \mid [\![\overrightarrow{Bob\ \ loves\ \ beer}]\!] \Big\rangle \ = \ \frac{1}{4}$$

$$\Big\langle [\![\overrightarrow{Bob\ \ does\ \ not\ \ like\ \ beer}]\!] \mid [\![\overrightarrow{Bob\ \ hates\ \ beer}]\!] \Big\rangle \ = \ \frac{3}{4}$$

$$\Big\langle [\![\overrightarrow{Bob\ \ does\ \ not\ \ like\ \ beer}]\!] \mid [\![\overrightarrow{Bob\ \ likes\ \ beer}]\!] \Big\rangle \ = \ \frac{3}{8}$$

Natural Language Meaning and Entanglement

Natural Language Meaning and Entanglement

Bob

Natural Language Meaning and Entanglement

Bob does

Natural Language Meaning and Entanglement

Bob does not

Natural Language Meaning and Entanglement

Bob does not like

Natural Language Meaning and Entanglement

Bob does not like beer.

Natural Language Meaning and Entanglement

Bob does not like beer.

like (Bob,beer)

Natural Language Meaning and Entanglement

Bob does not like beer.

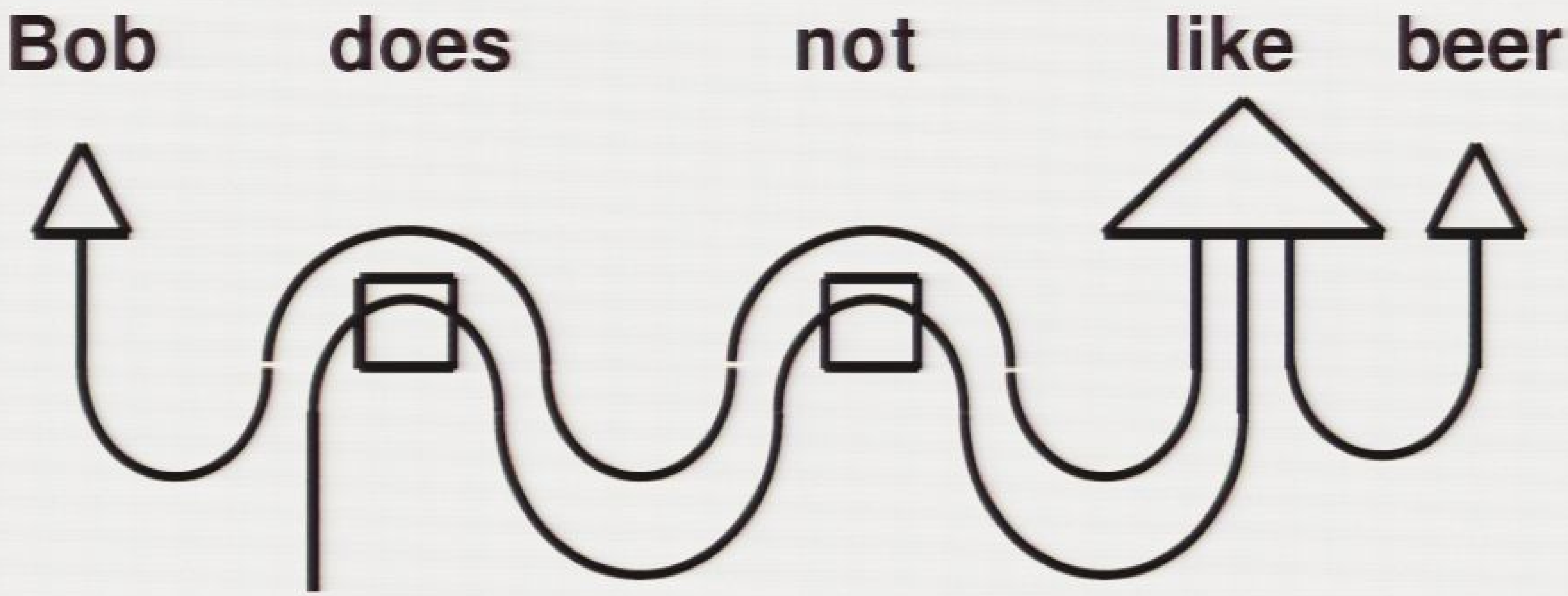
not $\circ$ like (Bob,beer)

Natural Language Meaning and Entanglement

Bob does not like beer.

$$\text{does} \circ \text{not} \circ \text{like}(\text{Bob}, \text{beer})$$

Natural Language Meaning and Entanglement



Passive vs Active Negation

As a result of our previous interpretation of "not"

"Bob does not like beer." $\cong$ "Bob dislikes beer.".

This is ok, but also that

"It is not the case that Bob likes beer." $\cong$ "Bob dislikes beer".

This is not ok: may be Bob is *indifferent* to beer or prefers whisky.

Two kinds of negation:

active in "*does not*" and passive in "*is not*".

Moving to 3 Dimensions

The passive negation is modeled in 3-dims by changing $\overrightarrow{s_{ij}}$ to

$$\overrightarrow{s}_{ij} = \begin{cases} |001\rangle & m_i \text{ likes } d_j \\ |100\rangle & m_i \text{ dislikes } d_j \\ |010\rangle & m_i \text{ is indifferent to } d_j \end{cases}$$

and define a new $\overrightarrow{not}$ and extend $\overrightarrow{does}$ to

$$\overrightarrow{pnot} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \qquad \overrightarrow{does} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Moving to 3 Dimensions

The active negation is modeled in 3-dims by defining a new $\overrightarrow{not}$ as

$$\overrightarrow{anot} = \begin{pmatrix} 0 & - & 1 \\ 0 & - & 0 \\ 1 & - & 0 \end{pmatrix}$$

As a result we have

$$\overrightarrow{anot}\,(|\,100\rangle) = |\,001\rangle, \qquad \overrightarrow{anot}\,(|\,001\rangle) = |\,100\rangle$$

If Bob does not like beer, then he dislikes it and vice versa.

So we have accommodate both kinds of negation by moving to 3-dims: active negation sends "likes" to "dislikes" similar to the Boolean case, passive negation sends each word to its orthogonal space, the disjunction of the other possibilities.

Summary

We have provided the first mathematical model for a compositional distributed model of meaning and stepped towards providing semantics for Pregroup grammars. We showed how string diagrams simplify our calculations to a great extent.

Assigning compositional vector meaning to a sentence has 3 steps:

1- Type the sentence using a Pregroup grammar.

2- Compute the vectors of the words therein: words with simple types as usual, words with compound types, as linear maps.

3- Apply to the tensor products of the meaning vectors of the words in the sentence the map of its syntactic reduction pre-composed with Bell states.

Moving to 3 Dimensions

As a result we have

$$\overrightarrow{pnot}\,(|\,100\rangle) = |\,011\rangle = |\,001\rangle + |\,010\rangle$$

Which makes the previous equality makes sense:

If it is not the case that Bob likes beer, then he should either dislike it or be indifferent to it.

Similarly we have

$$\overrightarrow{pnot}\,(|\,001\rangle) = |\,110\rangle = |\,100\rangle + |\,010\rangle$$

$$\overrightarrow{pnot}\,(|\,010\rangle) = |\,101\rangle = |\,100\rangle + |\,001\rangle$$

Moving to 3 Dimensions

The passive negation is modeled in 3-dims by changing $\overrightarrow{s_{ij}}$ to

$$\vec{s}_{ij} = \begin{cases} |001\rangle & m_i \text{ likes } d_j \\ |100\rangle & m_i \text{ dislikes } d_j \\ |010\rangle & m_i \text{ is indifferent to } d_j \end{cases}$$

and define a new $\overrightarrow{not}$ and extend $\overrightarrow{does}$ to

$$\overrightarrow{pnot} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \qquad \overrightarrow{does} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Passive vs Active Negation

As a result of our previous interpretation of ”not”

”Bob does not like beer.” $\cong$ ”Bob dislikes beer.”.

This is ok, but also that

”It is not the case that Bob likes beer.” $\cong$ ”Bob dislikes beer”.

This is not ok: may be Bob is *indifferent* to beer or prefers whisky.

Two kinds of negation:

active in ”*does not*” and passive in ”*is not*”.

Moving to 3 Dimensions

The passive negation is modeled in 3-dims by changing $\overrightarrow{s_{ij}}$ to

$$\vec{s}_{ij} = \begin{cases} |001\rangle & m_i \text{ likes } d_j \\ |100\rangle & m_i \text{ dislikes } d_j \\ |010\rangle & m_i \text{ is indifferent to } d_j \end{cases}$$

and define a new $\overrightarrow{not}$ and extend $\overrightarrow{does}$ to

$$\overrightarrow{pnot} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \qquad \overrightarrow{does} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Moving to 3 Dimensions

As a result we have

$$\overrightarrow{pnot}\,(|\,100\rangle) = |\,011\rangle = |\,001\rangle + |\,010\rangle$$

Which makes the previous equality makes sense:

If it is not the case that Bob likes beer, then he should either dislike it or be indifferent to it.

Similarly we have

$$\overrightarrow{pnot}\,(|\,001\rangle) = |\,110\rangle = |\,100\rangle + |\,010\rangle$$

$$\overrightarrow{pnot}\,(|\,010\rangle) = |\,101\rangle = |\,100\rangle + |\,001\rangle$$

Moving to 3 Dimensions

The active negation is modeled in 3-dims by defining a new $\overrightarrow{not}$ as

$$\overrightarrow{anot} = \begin{pmatrix} 0 & - & 1 \\ 0 & - & 0 \\ 1 & - & 0 \end{pmatrix}$$

As a result we have

$$\overrightarrow{anot}\,(|\,100\rangle) = |\,001\rangle, \qquad \overrightarrow{anot}\,(|\,001\rangle) = |\,100\rangle$$

If Bob does not like beer, then he dislikes it and vice versa.

So we have accommodate both kinds of negation by moving to 3-dims: active negation sends "likes" to "dislikes" similar to the Boolean case, passive negation sends each word to its orthogonal space, the disjunction of the other possibilities.

Summary

We have provided the first mathematical model for a compositional distributed model of meaning and stepped towards providing semantics for Pregroup grammars. We showed how string diagrams simplify our calculations to a great extent.

Assigning compositional vector meaning to a sentence has 3 steps:

1- Type the sentence using a Pregroup grammar.

2- Compute the vectors of the words therein: words with simple types as usual, words with compound types, as linear maps.

3- Apply to the tensor products of the meaning vectors of the words in the sentence the map of its syntactic reduction pre-composed with Bell states.

Work in Progress

The vectors of logical connectives such as ”and” ”or”, ”forall”, ”exists”, and their various natural language incarnations.

Implementing the system and run experiments to evaluate our compositional theory of meaning.

Applying the verified theory to Information Retrieval, e.g. from the web, or from more specialized texts such as those used in the court or patent offices.

Writing a UK EPSRC project among Oxford, Cambridge, Sussex, Edinburgh, York.