Classical Copying versus Quantum Entanglement in Natural Language: The Case of VP-ellipsis

Gijs Wijnholds
School of Electronic Engineering and Computer Science
Queen Mary University of London
g.j.wijnholds@qmul.ac.uk

Mehrnoosh Sadrzadeh
School of Electronic Engineering and Computer Science
Queen Mary University of London
mehrnoosh.sadrzadeh@qmul.ac.uk

This paper compares classical copying and quantum entanglement in natural language by considering the case of verb phrase (VP) ellipsis. VP ellipsis is a non-linear linguistic phenomenon that requires the reuse of resources, making it the ideal test case for a comparative study of different copying behaviours in compositional models of natural language. Following the line of research in compositional distributional semantics set out by [6] we develop an extension of the Lambek calculus which admits a controlled form of contraction to deal with the copying of linguistic resources. We then develop two different compositional models of distributional meaning for this calculus. In the first model, we follow the categorical approach of [5] in which a functorial passage sends the proofs of the grammar to linear maps on vector spaces and we use Frobenius algebras to allow for copying. In the second case, we follow the more traditional approach that one finds in categorial grammars, whereby an intermediate step interprets proofs as non-linear lambda terms, using multiple variable occurrences that model classical copying. As a case study, we apply the models to derive different readings of ambiguous elliptical phrases and compare the analyses that each model provides.

1 Introduction

Lexical distributional models of meaning assume that the meaning of a word is given by its context, an idea that can be operationalised by deriving vectorial word representations from corpus co-occurrence statistics. These single-word embeddings by now are an essential part of the computational linguistics toolkit, as they have been successfully applied in several NLP tasks (see e.g. [24, 4] for an overview). However, the move from single-word embeddings to larger phrases and full sentences poses a number of immediate issues. Firstly, the distributional hypothesis underlying the single-word embeddings does not directly apply to sentences: a sentence meaning is given by more than just its contexts. Second, such an approach would suffer from data sparsity: although there has been investigation in deriving vector representations for adjective-noun, verb-object, and subject-verb-object combinations from a corpus, e.g. see [16], current corpora are not able to provide enough examples to successfully represent phrases.

In this paper we discuss compositionality from the perspective of VP ellipsis, which constitutes phrases that lack a verb phrase component that however is often *marked* by another constituent in the phrase, often an auxiliary verb. Examples are "Mary drinks and Bob does too" and "Kim wears a hat but Sandy does not", in either case the auxiliary verb 'does' refers to the verb phrase occurring earlier in the sentence. VP ellipsis provides an interesting challenge for compositional distributional semantics for two reasons: first, it is a *non-linear phenomenon*, as the verb phrase is needed twice to parse the sentence and there is no straightforward linear algebraic way to deal with the reuse of resources. Secondly, one can easily extend the current experimental datasets of the setting [8, 14] and compute with it: VP ellipsis is only one step further from a simple transitive sentence. The examples above are both constructed

B. Coecke, J. Hedges, D. Kartsaklis, M. Lewis, D. Marsden (Eds.):
2018 Workshop on Compositional Approaches
for Physics, NLP, and Social Sciences (CAPNS)
EPTCS 283, 2018, pp. 103–119, doi:10.4204/EPTCS.283.8

using two subjects, an object, and a verb. This makes modelling VP ellipsis a suitable candidate for experimental evaluation[1].

Ellipsis has been modelled both as a syntactic and a semantic problem [15] and here we approach it from the perspective of categorical compositional distributional semantics [6, 5], in which the derivations of a typelogical grammar are interpreted as linear maps on vector spaces. The tight connection between syntax and semantics that one assumes in these models means that we will treat ellipsis as a phenomenon that requires controlled copying of resources in syntax. Thus, similar to the approaches of [10, 19, 20], we argue for the use of controlled forms of copying in a type logical system to deal with ellipsis. We then define two compositional architectures: a quantum entangled semantics following the direct categorical modelling of [5] where we use Frobenius algebras to interpret the copying operations, and a classical semantics in which an intermediate step allows for classical copying by means of variable reuse in terms of a non-linear lambda calculus.

This paper is structured as follows: in Section 2, we discuss the problem of ellipsis and argue for the legitimacy of non-linearity in the syntactic process. We define an extension of the Lambek calculus to deal with resource reuse in syntax in Section 3. In Sections 4 and 5, respectively, we instantiate this calculus to a categorical and a classical model. We then apply these two different semantics to derive different readings of elliptical phrases with structural ambiguities and conclude in Section 6.

2 Background

Loosely following [7, 11] we define ellipsis as a phenomenon in which two phrases are parallel in structure, though one of the phrases is incomplete and requires material from the other phrase to be *copied* in order to make sense. In the case of verb phrase ellipsis there is usually a *marker* present that specifies that material needs to be copied and moved into place. In the example of verb phrase ellipsis in Equation 1, where the elided verb phrase is marked by the auxiliary verb. Ideally, sentence 1(a) is in a bidirectional entailment relation with 1(b), i.e. (a) entails (b) and (b) entails (a).

$$\begin{array}{ll} a & \text{“Alice drinks and Bill does too”} \\ b & \text{“Alice drinks and Bill drinks”} \end{array} \tag{1}$$

A more complicated example of ellipsis and anaphora, that induces an ambiguity, is the one in Equation 2, where the ambiguous phrase (a) has two readings (b) and (c).

$$\begin{array}{lll} a & \text{“Gary loves his code and Bill does too”} & \text{(ambiguous)} \\ b & \text{“Gary loves Gary's code and Bill loves Gary's code”} & \text{(strict)} \\ c & \text{“Gary loves Gary's code and Bill loves Bill's code”} & \text{(sloppy)} \end{array} \tag{2}$$

In a formal semantics account, the first example could be analysed with the auxiliary verb as an identity function on the main verb of the sentence and an intersective meaning for the coordinator. Somehow the parts need to be appropriately combined to produce the reading (b) for sentence (a):

$$\begin{array}{c} \text{does too} : \lambda x.x \\ \text{and} : \lambda x.\lambda y.(x \wedge y) \end{array} \quad \text{should give} \quad \mathbf{drinks}(\mathbf{alice}) \wedge \mathbf{drinks}(\mathbf{bill})$$

The second example would assume the same meaning for the coordinator and auxiliary but now the possessive pronoun “his” gets a more complicated term: $\lambda x.\lambda y.\mathbf{owns}(x,y)$. The analysis then somehow should derive two readings:

[1] This work is currently under review and could therefore not be included here as of yet.

$$\mathbf{loves}(\mathbf{gary},x) \wedge \mathbf{owns}(\mathbf{gary},x) \wedge \mathbf{loves}(\mathbf{bill},x) \quad \text{(strict)}$$

$$\mathbf{loves}(\mathbf{gary},x) \wedge \mathbf{owns}(\mathbf{gary},x) \wedge \mathbf{loves}(\mathbf{bill},y) \wedge \mathbf{owns}(\mathbf{bill},y) \quad \text{(sloppy)}$$

There are three issues with these analyses that a distributional semantic treatment should address: first, one has to determine how individual word meanings are combined to form the phrase meaning (syntax). Second, function words such as the coordinator 'and' need to be given a lexical meaning since they should not be addressed distributionally (lexical semantics). Thirdly, the semantic representations are all non-linear: the main verb is used twice in the first example, the noun phrases are consumed twice or thrice in the second example. Somehow a model needs to account for how these non-linearities are obtained (derivational semantics).

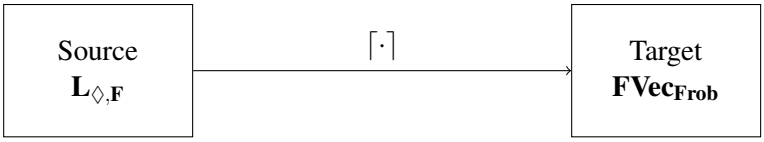
To address the first issue we define an extension of the Lambek calculus that allows for the copying of resources by means of a controlled proof-theoretic contraction rule. The lexical semantics of the coordinator ('and') and the possessive pronoun ('his') are given by using Frobenius algebras. This approach was shown to be fruitful in previous work [12, 22]. We discuss the third issue below.

Quantum versus Classical non-linearity To set up a compositional distributional model that allows a certain non-linearity there is a choice to be made: to stay within the existing categorical framework [6, 5], we want to work with compact closed categories and their concrete instantiation to the category of vector spaces and linear maps. The Frobenius algebras that have been used in previous work to deal with relative pronouns and coordination, can also be used to allow the copying of resources, however this will happen in a non-cartesian way in which the material that was copied, is entangled. The main Frobenius map that is used for relative pronouns by [22, 18] expresses element wise multiplication, but its dual map copies a vector by placing its values on the diagonal of a square matrix. In terms of a type signature this indeed multiplies the vector space on which the map is performed, but does not allow for the actual vector to be used in a non-entangled way. In fact, there is no linear map that can copy arbitrary vectors in the cartesian sense [1, 9]. To see this for a concrete example, consider the phrase "Alice loves herself" with tensors $\mathbf{alice} = \sum_i a_i \vec{v}_i$, and $\mathbf{loves} = \sum_{jkl} c_{jkl}(\vec{v}_j \otimes \vec{s}_k \otimes \vec{v}_l)$. The interpretation of a classical semantics (left) differs from the result of using Frobenius algebras (right):

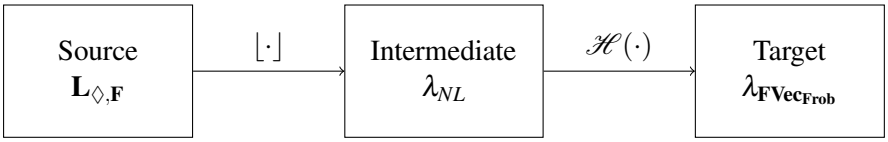
Classical	**Frobenius**
$\mathbf{alice}_i\mathbf{loves}_{ijk}\mathbf{alice}_k = \sum_{ijk} a_i c_{ijk} a_k \vec{s}_j$	$\mathbf{alice}_i\mathbf{loves}_{iji} = \sum_{ij} a_i a_i c_{iji} \vec{s}_j$

Within the categorical framework we could derive the sentence meaning on the right, but not the one on the left. However, we can define a different compositional distributional model that allows for the cartesian copying to take place. Since we do not want to fix this choice in advance, we define both models and give a comparison in their treatment of VP ellipsis to see whether a classical non-linearity is preferred over am entangled non-linearity.

Two Architectures The categorical framework implements compositionality directly as a functorial passage from a syntactic category to a semantic category. The concrete model of [5] takes the Lambek calculus as a monoidal biclosed category, which then is mapped onto the category of vector spaces and linear maps, where the tensor product is monoidal and whose internal hom is given by the space of maps between two vector spaces. For our case, we take an extension of the Lambek calculus with controlled contraction which we will denote by $\mathbf{L}_{\Diamond,\mathbf{F}}$, and map it onto the category of vector spaces where each space has a Frobenius algebra, written $\mathbf{FVec}_{\mathbf{Frob}}$. In a picture, the process looks like



In order to allow classical copying behaviour in a compositional distributional model, we decompose the categorical model of [5] into a two-step architecture: derivations and are now mapped onto terms of a non-linear simply typed lambda calculus λ_{NL}. The second stage of the interpretation process replaces the assumed lexical constants for words by their lexical semantics, finally resulting in a term of a lambda calculus that models vectors and linear maps, denoted $\lambda_{\mathbf{FVec}_{\mathbf{Frob}}}$. In a picture:



The effect is that we allow the cartesian behaviour of copying elements before concretisation in a vector semantics: the meaning of a sentence now is a program that has non-linear access to word embeddings.

3 A Proof System for Controlled Copying

Our starting point for syntax is the Lambek Calculus, the noncommutative fragment of multiplicative linear logic without units. Formulas in F are built from a set of basic formulas B and using the connectives $\otimes, \backslash, /$, sharing the residuation relation[2] expressed in Figure 2. Moreover, we add the control modalities $\Diamond, \Box$. The modalities have a purely syntactical role: instead of directly allowing the copying of resources, the system is designed such that only a type that is labelled with a $\Diamond$ can be copied. This prevents the overgeneration of a general contraction rule, but allows the copying of those words that have been decorated with a $\Diamond$ type. The residuated $\Box$ modality allows for the system to operate on the rest of a $\Diamond$ decorated type without losing track of the position of the $\Diamond$. Combining $\Diamond$ with a $\backslash$ or $/$ creates the behaviour wanted for ellipsis: an ellipsis marker will generally be annotated with a type $\Diamond A \backslash B$, meaning that it expects a copy of a resource of type A *somewhere* to its left. Once the copy is created, it is moved into the right position to be consumed by the ellipsis marker, as expressed in Figure 1. To this end, the modalities license access to (limited forms of) contraction and commutativity, through the use of structural rules (see Figure 2). A similar setup has shown to give an account for pronoun relativisation [17], which in the context of distributional semantics has been worked out in [22, 18].

The structural rules can also be stipulated in equivalent axiomatic form, but for the purpose of parsing it is more useful to consider rule form. We want to have a system that enjoys decidability: though this is not a straightforward property in the presence of just contraction[3], our system becomes decidable easily if we put a bound on the number of contractions in a proof.

In order to determine the analysis of a phrase, we have a mapping $\sigma : \Sigma \to F$ where Σ is a list of words. Sentencehood of a sequence of words $w_1, ..., w_n$ is then determined by derivability with respect to a distinguished sentence type s of the conjoined formula $\sigma(w_1) \otimes ... \otimes \sigma(w_n)$, and this generalises to arbitrary goal formulas. In other words, whenever $\sigma(w_1) \otimes ... \otimes \sigma(w_n) \to A$ is derivable, we may say that $w_1, ..., w_n$ is a phrase *of type* A.

[2] Algebraically, three operations $\cdot, \leftarrow, \rightarrow$ on a partial order form a residuated triple iff $a \leq c \leftarrow b \Leftrightarrow a \cdot b \leq c \Leftrightarrow b \leq a \rightarrow c$. Logically, this corresponds to the two-way inference rules that we use here.

[3] See the discussion in Katalin Bimbó's monograph [2] and the results of [23, 3].

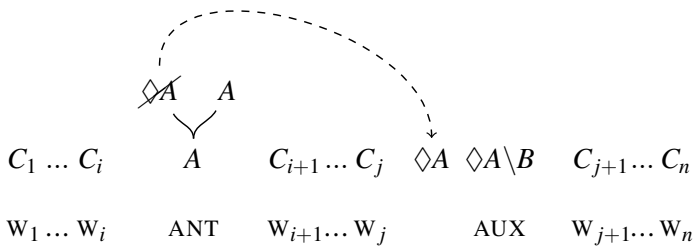


Figure 1: General strategy for ellipsis resolution in $\mathbf{L}_{\Diamond,\mathbf{F}}$. The antecedent is copied, and the $\Diamond$ decorated copy is moved directly left of the marker which consumes the copy.

$$\frac{}{1_A : A \longrightarrow A} \qquad \frac{f : A \longrightarrow B \quad g : B \longrightarrow C}{g \circ f : A \longrightarrow C}$$

$$\frac{f : \Diamond A \longrightarrow B}{\triangledown f : A \longrightarrow \Box B} \qquad \frac{f : A \otimes B \longrightarrow C}{\triangleright f : A \longrightarrow C/B} \qquad \frac{f : A \otimes B \longrightarrow C}{\triangleleft f : B \longrightarrow A\backslash C}$$

$$\frac{g : A \longrightarrow \Box B}{\triangledown^{-1} g : \Diamond A \longrightarrow B} \qquad \frac{g : A \longrightarrow C/B}{\triangleright^{-1} g : A \otimes B \longrightarrow C} \qquad \frac{g : B \longrightarrow A\backslash C}{\triangleleft^{-1} g : A \otimes B \longrightarrow C}$$

$$\frac{f : (A \otimes B) \otimes C \longrightarrow D}{\widehat{\alpha}^{l}_{\Diamond}(f) : A \otimes (B \otimes C) \longrightarrow D} \qquad \frac{f : A \otimes (B \otimes C) \longrightarrow D}{\widehat{\alpha}^{r}_{\Diamond}(f) : (A \otimes B) \otimes C \longrightarrow D}$$

$$\frac{f : \Diamond A \otimes A \longrightarrow B}{\widehat{C}(f) : A \longrightarrow B} \qquad \frac{f : A \otimes (\Diamond B \otimes C) \longrightarrow D}{\widehat{M}(f) : (\Diamond B \otimes A) \otimes C \longrightarrow D} \qquad \frac{f : \Diamond A \otimes (\Diamond B \otimes C) \longrightarrow D}{\widehat{S}(f) : \Diamond B \otimes (\Diamond A \otimes C) \longrightarrow D}$$

Figure 2: $\mathbf{L}_{\Diamond,F}$. Residuation rules; Structural postulates for controlled copying and moving (rule form). The names of the rules are given by the term symbols in the conclusion of each rule.

3.1 Deriving Ellipsis

Let's consider again the examples given in Section 2. An elliptical phrase like "Alice drinks and Bob does too" requires the verb phrase to be used twice; the proof system handles this by means of the structural rules of controlled contraction and movement. Figure 3 shows the short-hand derivation for the phrase — skipping trivial applications of associativity — with the copy of the verb phrase highlighted in red. The derivation follows the general pattern depicted in Figure 1, with the auxiliary verb "does too" marking the ellipsis site and therefore typed $\Diamond(np\backslash s)\backslash(np\backslash s)$; requiring a copied verb from *somewhere* to its left, it will return a verb. Reading the proof from bottom to top, first a contraction is applied to copy the verb phrase, marking the copy with a control modality. Then, the copy of the verb phrase is structurally moved rightward until it is in the right position to interact with the auxiliary ellipsis marker. It is not hard to see, then, that this allows the meaning of the verb phrase to be multiplied, have the copied version interact with the auxiliary to give the meaning of the whole phrase in a similar way as the meaning of the

expanded phrase “Alice drinks and Bob drinks” would be computed.

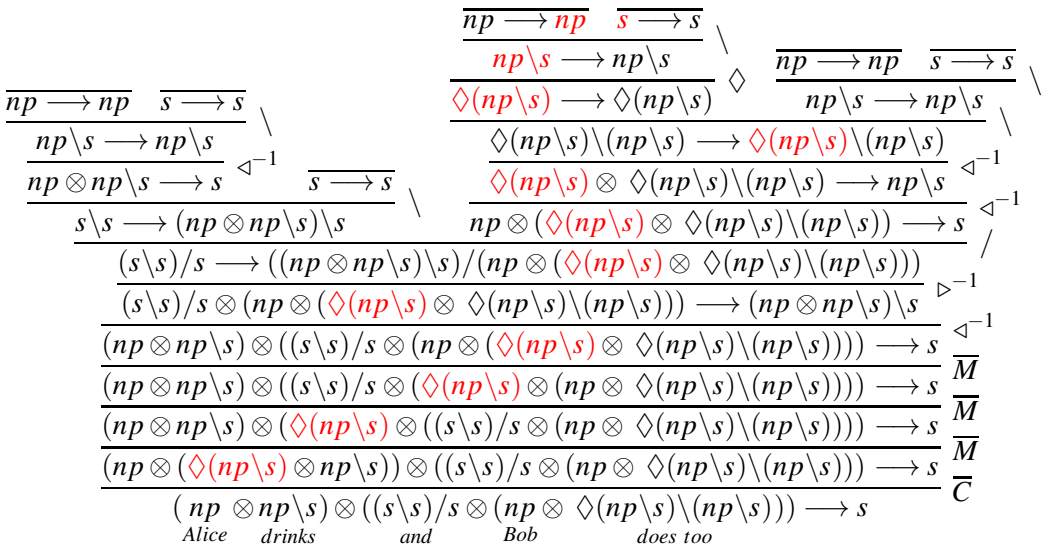


Figure 3: Short hand derivation for “Alice drinks and Bob does-too”. The copied verb and its subformulas are highlighted in red. The bottom inferences copy the verb and move it to the auxiliary verb (structural phase), after which logical rules are applied to reduce to axiom leaves (logical phase).

More complicated cases of ellipsis contain anaphora, and allow an interaction that leads to strict and sloppy readings. To analyse the examples from Equation 1, the given proof system deals with the ambiguity by allowing a choice of resolution: either the anaphora gets resolved first, after which the resolved verb phrase is copied (strict reading), or the unresolved verb phrase is copied and resolved once for each subclause (sloppy reading). The two readings are derived in Figures 4, 5, 6. In the following two sections, we give respectively a quantum and a classical interpretation to these derivations.

4 Quantum Semantics with Frobenius Algebras

In order to derive the semantics for coordination and ellipsis proposed in [12, 13], we follow the framework of [5] and label the rules of our proof system with categorical morphisms in a compact closed category endowed with Frobenius algebras. Recall that a *compact closed category* (CCC) is a monoidal category, i.e. it has an associative $\otimes$ with unit I, and for every object there is a left and a right adjoint with maps

$$A^l \otimes A \xrightarrow{\varepsilon^l} I \xrightarrow{\eta^l} A \otimes A^l \qquad A \otimes A^r \xrightarrow{\varepsilon^r} I \xrightarrow{\eta^r} A^r \otimes A$$

These need satisfy the so-called yanking equations

$$(1_A \otimes \varepsilon^l_A) \circ (\eta^l_A \otimes 1_A) = 1_A \qquad (\varepsilon^r_A \otimes 1_A) \circ (1_A \otimes \eta^r_A) = 1_A$$

$$(\varepsilon^l_A \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l_A) = 1_{A^l} \qquad (1_{A^r} \otimes \varepsilon^r_A) \circ (\eta^r_A \otimes 1_{A^r}) = 1_{A^r}$$

In a *symmetric* CCC, the tensor moreover is commutative, and we can write A^* for the collapsed left and right adjoints. Frobenius algebras are defined over the objects of a category, and we say that an object A has a Frobenius structure when it has maps

$$\Delta : A \to A \otimes A \quad \mu : A \otimes A \to A \quad \iota : A \to I \quad \zeta : I \to A$$

satisfying monoidality and comonoidality for the pairs (A, μ, ζ) and (A, Δ, ι) and the Frobenius equation

$$(\mu \otimes id_A) \circ (id_A \otimes \delta) = \delta \circ \mu = (id_A \otimes \mu) \circ (\delta \otimes id_A)$$

In the concrete instance of **FVect**, the unit I stands for the field $\mathbb{R}$; identity maps, composition and tensor product are defined as usual. Since bases of vector spaces are fixed in concrete models, there is only one natural way of defining a basis for a *dual space*, so that $V^* \cong V$. In concrete models we may collapse the adjoints completely. The ε map takes inner products, whereas the η map (with $\lambda = 1$) introduces an identity tensor as follows:

$$\begin{array}{llccl} \varepsilon_V : V \otimes V \to \mathbb{R} & \text{given by} & \sum_{ij} v_{ij}(\vec{e}_i \otimes \vec{e}_j) & \mapsto & \sum_i v_{ii} \\ \eta_V : \mathbb{R} \to V \otimes V & \text{given by} & \lambda & \mapsto & \sum_i \lambda(\vec{e}_i \otimes \vec{e}_i) \end{array}$$

Any finite vector space with fixed basis possesses a Frobenius structure, and so we write $\mathbf{FVect}_{\mathbf{Frob}}$ for the category of finite dimensional vector spaces with fixed basis[4]. The Frobenius maps take the form given below: Δ takes a tensor and places its values on the diagonal of a square matrix, whereas μ extracts the diagonal from a square matrix. The ι and ζ maps respectively sum the coefficients of a vector or introduce a vector with the value 1 for all of its coefficients.

$$\begin{array}{llccl} \Delta_V : V \to V \otimes V & \text{given by} & \sum_i v_i \vec{e}_i & \mapsto & \sum_i v_i(\vec{e}_i \otimes \vec{e}_i) \\ \iota_V : V \to \mathbb{R} & \text{given by} & \sum_i v_i \vec{e}_i & \mapsto & \sum_i v_i \\ \mu_V : V \otimes V \to V & \text{given by} & \sum_{ij} v_{ij}(\vec{e}_i \otimes \vec{e}_j) & \mapsto & \sum_i v_{ii} \vec{e}_i \\ \zeta_V : \mathbb{R} \to V & \text{given by} & \lambda & \mapsto & \sum_i \lambda \vec{e}_i \end{array}$$

4.1 Interpretation: Proofs and Morphisms

The interpretation of derivations as linear maps has two components: on the type level, formulas are associated with vector spaces. On the proof level, the abstract terms of a proof become operations on vector spaces that respect the type interpretation. On basic types, the interpretation $\lceil \cdot \rceil$ assigns arbitrary vector spaces, on complex types we have

$$\lceil A \otimes B \rceil = \lceil A \rceil \otimes \lceil B \rceil \quad \lceil A/B \rceil = \lceil A \rceil \otimes \lceil B \rceil^* \quad \lceil A \backslash B \rceil = \lceil A \rceil^* \otimes \lceil B \rceil \quad \lceil \Diamond A \rceil = \lceil \Box A \rceil = \lceil A \rceil$$

On the level of proofs, we follow the standard interpretation given in [5, 25] to interpret the basic logic of residuation, and add the interpretation of the control rules from Figure 2. The simplest interpretations are identity and composition: $\lceil 1_A \rceil = 1_{\lceil A \rceil}$, $\lceil g \circ f \rceil = \lceil g \rceil \circ \lceil f \rceil$. For the residuation inferences, we take the map $\lceil f \rceil : \lceil A \rceil \otimes \lceil B \rceil \longrightarrow \lceil C \rceil$ interpreting the premise, and define

$$\lceil \triangleright f \rceil = \lceil A \rceil \xrightarrow{1_{\lceil A \rceil} \otimes \eta_{\lceil B \rceil}} \lceil A \rceil \otimes \lceil B \rceil \otimes \lceil B \rceil^* \xrightarrow{\lceil f \rceil \otimes 1_{\lceil B \rceil^*}} \lceil C \rceil \otimes \lceil B \rceil^*$$

$$\lceil \triangleleft f \rceil = \lceil B \rceil \xrightarrow{\eta_{\lceil A \rceil} \otimes 1_{\lceil B \rceil}} \lceil A \rceil^* \otimes \lceil A \rceil \otimes \lceil B \rceil \xrightarrow{1_{\lceil A \rceil^*} \otimes \lceil f \rceil} \lceil A \rceil^* \otimes \lceil C \rceil$$

[4] In concrete experimental models bases are always fixed.

For the inverses, from maps $\lceil g \rceil : \lceil A \rceil \longrightarrow \lceil C/B \rceil$, $\lceil h \rceil : \lceil B \rceil \longrightarrow \lceil A \backslash C \rceil$ for the premises, we define

$$\lceil \triangleright^{-1} g \rceil = \lceil A \rceil \otimes \lceil B \rceil \xrightarrow{\lceil g \rceil \otimes 1_{\lceil B \rceil}} \lceil C \rceil \otimes \lceil B \rceil^* \otimes \lceil B \rceil \xrightarrow{1_{\lceil C \rceil} \otimes \varepsilon_{\lceil B \rceil}} \lceil C \rceil$$

$$\lceil \triangleleft^{-1} h \rceil = \lceil A \rceil \otimes \lceil B \rceil \xrightarrow{1_{\lceil A \rceil} \otimes \lceil h \rceil} \lceil A \rceil \otimes \lceil A \rceil^* \otimes \lceil C \rceil \xrightarrow{\varepsilon_{\lceil A \rceil} \otimes 1_{\lceil C \rceil}} \lceil C \rceil$$

For the (derived) rules of monotonicity, the case of parallel composition is immediate: $\lceil f \otimes g \rceil = \lceil f \rceil \otimes \lceil g \rceil$. For the slash cases, from $\lceil f \rceil : \lceil A \rceil \longrightarrow \lceil B \rceil$ and $\lceil g \rceil : \lceil C \rceil \longrightarrow \lceil D \rceil$, we obtain

$$\lceil f/g \rceil = \qquad\qquad \lceil f \backslash g \rceil =$$

$$\begin{array}{rl} \lceil A \rceil \otimes \lceil D \rceil^* & \\ \downarrow \lceil f \rceil \otimes \eta_{\lceil C \rceil} \otimes 1_{\lceil D \rceil^*} & \\ \lceil B \rceil \otimes \lceil C \rceil^* \otimes \lceil C \rceil \otimes \lceil D \rceil^* & \\ \downarrow 1_{\lceil B \rceil \otimes \lceil C \rceil^*} \otimes \lceil g \rceil \otimes 1_{\lceil D \rceil^*} & \\ \lceil B \rceil \otimes \lceil C \rceil^* \otimes \lceil D \rceil \otimes \lceil D \rceil^* & \\ \downarrow 1_{\lceil B \rceil \otimes \lceil C \rceil^*} \otimes \varepsilon_{\lceil D \rceil} & \\ \lceil B \rceil \otimes \lceil C \rceil^* & \end{array} \qquad \begin{array}{rl} \lceil B \rceil^* \otimes \lceil C \rceil & \\ \downarrow 1_{\lceil B \rceil^*} \otimes \eta_{\lceil A \rceil} \otimes \lceil g \rceil & \\ \lceil B \rceil^* \otimes \lceil A \rceil \otimes \lceil A \rceil^* \otimes \lceil D \rceil & \\ \downarrow 1_{\lceil B \rceil^*} \otimes \lceil f \rceil \otimes 1_{\lceil A \rceil^* \otimes \lceil D \rceil} & \\ \lceil B \rceil^* \otimes \lceil B \rceil \otimes \lceil A \rceil^* \otimes \lceil D \rceil & \\ \downarrow \varepsilon_{\lceil B \rceil} \otimes 1_{\lceil A \rceil^* \otimes \lceil D \rceil} & \\ \lceil A \rceil^* \otimes \lceil D \rceil & \end{array}$$

Interpretation for the associativity structural rules is immediate via the standard associativity of **FVect**: $\lceil \widehat{\alpha}^l_\diamond f \rceil = \lceil f \rceil \circ \alpha^{-1}$ and $\lceil \widehat{\alpha}^r_\diamond f \rceil = \alpha \circ \lceil f \rceil$. For the other structural rules, we additionally use the symmetry maps of **FVect** as well as the diagonal Frobenius map Δ:

$$\lceil \widehat{C}(f) \rceil = \lceil A \rceil \xrightarrow{\Delta_{\lceil A \rceil}} \lceil A \rceil \otimes \lceil A \rceil \xrightarrow{\lceil f \rceil} \lceil B \rceil$$

$$\lceil \widehat{M}(f) \rceil = (\lceil B \rceil \otimes \lceil A \rceil) \otimes \lceil C \rceil \xrightarrow{\sigma_{\lceil B \rceil, \lceil A \rceil} \otimes 1_{\lceil C \rceil}} (\lceil A \rceil \otimes \lceil B \rceil) \otimes \lceil C \rceil \xrightarrow{\alpha} \lceil B \rceil \otimes (\lceil A \rceil \otimes \lceil C \rceil) \xrightarrow{\lceil f \rceil} \lceil D \rceil$$

$$\lceil \widehat{S}(f) \rceil = \lceil B \rceil \otimes (\lceil A \rceil \otimes \lceil C \rceil) \xrightarrow{\alpha^{-1}} (\lceil B \rceil \otimes \lceil A \rceil) \otimes \lceil C \rceil \xrightarrow{\sigma_{\lceil B \rceil, \lceil A \rceil} \otimes 1_{\lceil C \rceil}} (\lceil A \rceil \otimes \lceil B \rceil) \otimes \lceil C \rceil \xrightarrow{\lceil f \rceil \circ \alpha} D$$

4.2 Frobenius Semantics of Ellipsis

Setting $\lceil np \rceil = N, \lceil s \rceil = S$, the proof in Figure 3 will be mapped on a morphism

$$N \otimes (N \otimes S) \otimes (S \otimes S \otimes S) \otimes N \otimes (N \otimes S \otimes N \otimes S) \xrightarrow{F} S$$

Let us write $[f]$ for a morphism which contains f but is surrounded by identity morphisms, and let us index it $[f_{A_i}]$ to specify the map acts on the ith occurrence of the object A in an object, needed whenever there may be an ambiguity. Then, we can write the full morphism as

$$F = [\varepsilon_N \otimes \varepsilon_S \otimes id_S \otimes \varepsilon_S] \circ [\varepsilon_{N_3}] \circ [\varepsilon_{N_4 \otimes S_5}] \circ [\sigma_{N_3 \otimes S_5, N_4}] \circ [\sigma_{N_3 \otimes S_2, S_3 \otimes S_4 \otimes S_5}] \circ [\sigma_{N_2 \otimes S_1, N_3 \otimes S_2}] \circ [\Delta_{N_2 \otimes S_1}]$$

That is, the Frobenius copying map is applied to the content of the verb tensor, after which the 'copy' is then moved to the right into the position next to the space $N \otimes S \otimes N \otimes S$ in which the auxiliary expression lives. We then perform contractions in the expected way. For concrete maps we will write $\times$ for the tensor contraction defined in the previous section, and $\cdot^{\perp}$ for the transpose of a tensor; $\odot$ denotes element wise multiplication. We can use boldface for concrete tensors to get the final concrete map

$$\mathbf{alice} \otimes \mathbf{drinks} \otimes \mathbf{and} \otimes \mathbf{bob} \otimes \mathbf{does\ too} \mapsto (\mathbf{alice} \odot \mathbf{bob})^{\perp} \times \mathbf{drinks}$$

And similarly, for a transitive case, one would obtain the meaning

$$(\mathbf{alice} \odot \mathbf{bob})^{\perp} \times (\mathbf{drinks} \times \mathbf{beer})$$

For the more involved examples of Figures 5, 6 we obtain more complicated maps that make a choice in the order of resolution of the ellipsis and the anaphoric reference. Rewriting the obtained linear map using the 'spider' equation of a commutative special Frobenius algebra, the normal form for the strict derivation, in which Gary fulfills the role of the anaphora before the ellipsis is resolved, gives

$$\mathbf{gary} \otimes \mathbf{loves} \otimes \mathbf{his} \otimes \mathbf{code} \otimes \mathbf{and} \otimes \mathbf{bob} \otimes \mathbf{does\ too} \mapsto \mu_N(\mathbf{gary} \odot \mathbf{bob} \odot \mathbf{code})_{ik}\mathbf{loves}_{ijk}$$

That is, in the final result the two subjects and the object are multiplied element wise, after which the resulting vector is expanded to be consumed by the verb. Note that the meaning of the phrase "Gary loves Gary's code and Bob loves Gary's code" is thus the same as the meaning of "Gary loves Bob's code and Bob loves Bob's code", due to the fact that we instantiated the anaphoric element 'his' with element wise multiplication.

For the sloppy derivation ("Gary loves Gary's code and Bob loves Bob's code") the situation is more problematic; the normal form for the meaning of the sloppy phrase is identical to the strict reading:

$$\mathbf{gary} \otimes \mathbf{loves} \otimes \mathbf{his} \otimes \mathbf{code} \otimes \mathbf{and} \otimes \mathbf{bob} \otimes \mathbf{does\ too} \mapsto \mu_N(\mathbf{gary} \odot \mathbf{bob} \odot \mathbf{code})_{ik}\mathbf{loves}_{ijk}$$

So not only do the readings have a symmetric meaning, the two different readings get the same semantics. The only way to mend this would be to change the lexical meaning of the anaphora 'his' for an operator other than element wise multiplication, though we suspect similar problems will arise.

5 Classical Semantics with Lambdas and Tensors

In this section, we develop a two-step semantics: first, we map the sentences to a non-linear lambda calculus, and from there to vectors and tensors. The meaning of a sentence now is a program with non-linear access to word embeddings. As depicted in the diagrammatic plan in Section 2, we deploy the same proof theory as in the one-step setup above, but rather than mapping types to vector spaces and proofs to linear maps, we map respectively to types and terms of a non-linear lambda calculus, after which a second translation steps replaces lexical constants associated with words by (terms modelling) concrete vectors and linear maps. In the next two subsections we work out these two steps.

5.1 Derivational Semantics: Lambdas and Constants

In the first step of the interpretation process, we map types and proofs onto types and terms of a non-linear simply typed lambda calculus with products. Similar to subsection 5.2, we define an arbitrary interpretation on basic types $\lfloor . \rfloor$, and define on complex types

$$\lfloor A \otimes B \rfloor = \lfloor A \rfloor \times \lfloor B \rfloor \quad \lfloor A/B \rfloor = \lfloor A \rfloor \to \lfloor B \rfloor \quad \lfloor A \backslash B \rfloor = \lfloor A \rfloor \to \lfloor B \rfloor \quad \lfloor \Diamond A \rfloor = \lfloor \Box A \rfloor = \lfloor A \rfloor$$

For the proofs, we can interpret identity and composition straightforwardly by $\lambda x.x$ and $\lambda x.N\ (M\ x)$ for M and N the terms of the subproofs, respectively. The binary residuation rules correspond to application and abstraction depending on the direction in which the rule is applied, whereas unary residuation does not change the terms at all

$$\lfloor \triangleright M \rfloor = \lambda x\ y.M\ \langle x,y\rangle \qquad \lfloor \triangleleft M \rfloor = \lambda y\ x.M\ \langle x,y\rangle \qquad \lfloor \triangledown M \rfloor = M$$
$$\lfloor \triangleright^{-1} N \rfloor = \lambda \langle x,y\rangle.(N\ x)\ y \qquad \lfloor \triangleleft^{-1} N \rfloor = \lambda \langle x,y\rangle.(N\ y)\ x \qquad \lfloor \triangledown^{-1} N \rfloor = N$$

The derived monotonicity rules get the interpretation below:

$$\lfloor M \otimes N \rfloor = \lambda \langle x,y\rangle.\langle M\ x, N\ y\rangle \qquad \lfloor M \backslash N \rfloor = \lambda f\ x.N\ (f\ (M\ x)) \qquad \lfloor M / N \rfloor = \lambda f\ x.M\ (f\ (N\ x))$$

The associativity rules behave as an identity since associativity is implicit in lambda terms.

$$\lfloor \widehat{\alpha}^{l}_{\diamond}(M) \rfloor = \lambda \langle x,y,z\rangle.M\ \langle x,y,z\rangle \qquad \lfloor \widehat{\alpha}^{r}_{\diamond}(M) \rfloor = \lambda \langle x,y,z\rangle.M\ \langle x,y,z\rangle$$

The non-linear behaviour enters with the interpretation for the structural rules for movement and copying:

$$\lfloor \widehat{C}(M) \rfloor = \lambda x.M\ \langle x,x\rangle \qquad \lfloor \widehat{M}(M) \rfloor = \lambda \langle y,x,z\rangle.M\ \langle x,y,z\rangle \qquad \lfloor \widehat{S}(M) \rfloor = \lambda \langle y,x,z\rangle.M\ \langle x,y,z\rangle$$

In this first step of interpretation, the words of a phrase are assigned constants in a term that is built up by translating the proof term using above translation. For the sample derivation of Figure 3, the logical phase computes only reductions; the subproof of the type

$$(np \otimes np\backslash s) \otimes ((s\backslash s)/s \otimes (np \otimes (\Diamond(np\backslash s) \otimes \Diamond(np\backslash s)\backslash(np\backslash s)))) \longrightarrow s$$

gives an abstract term

$$\lambda \langle \mathtt{subj}_1, \mathtt{verb}, \mathtt{coord}, \mathtt{subj}_2, \mathtt{verb}^*, \mathtt{aux}\rangle.(\mathtt{coord}\ ((\mathtt{aux}\ \mathtt{verb}^*)\ \mathtt{subj}_2))(\mathtt{verb}\ \mathtt{subj}_1)$$

and the structural phase repositions the copy $\mathtt{verb}^*$ next to the verb, after which the contraction rule *identifies* the variables associated with them, unifying $\mathtt{verb}$ and $\mathtt{verb}^*$:

$$\lambda \langle \mathtt{subj}_1, \mathtt{verb}, \mathtt{coord}, \mathtt{subj}_2, \mathtt{aux}\rangle.(\mathtt{coord}\ ((\mathtt{aux}\ \mathtt{verb})\ \mathtt{subj}_2))(\mathtt{verb}\ \mathtt{subj}_1)$$

We get the final *abstract proof term* for the proof in Figure 3 by applying the term above to the constants for the words in the sentence:

$$(\mathtt{and}\ ((\mathtt{dt}\ \mathtt{drinks})\ \mathtt{bob}))(\mathtt{drinks}\ \mathtt{alice}) : s \tag{3}$$

5.2 Lexical Semantics: Lambdas and Tensors

We complete the vector semantics by adding the second step in the interpretation process, which is the insertion of lexical entries for the assumptions/constants occurring in a proof term. In this step we face the issue that interpretation directly into a vector space is not an option given that there is no copying map that is linear, while at the same time lambda terms don't seemingly reflect vectors. However, following [21] we can model vectors using a lambda calculus as shown in the subsection below.

5.2.1 Lambdas and Tensors

Vectors can be seen as functions from natural numbers to the values in the underlying field, allowing us to represent them naturally as lambda terms. For any dimensionality n, we assume a basic type I_n, representing a finite index set (in concrete models the number of index types will be finite). The underlying field, in our case the real numbers $\mathbb{R}$, is given by the type R.

The type of a vector in $\mathbb{R}^n$ is now $V^n = I_n \to R$, the type of an $n \times m$ matrix is $M^{n\times m} = I_n \to I_m \to R$. In general, we may represent an arbitrary tensor with dimensions $n, m, ..., p$ by $T^{n\times m...\times p} = I_n \to I_m \to ... \to I_p \to R$. We will leave out the superscripts denoting dimensionality when they are either irrelevant or understood from the context.

By reference to index notation for linear algebra, we write $v\ i$ as v_i whenever it is understood that i is of type I. We moreover assume constants for the basic operations of a vector space: $0 : R, 1 : R, + : R \to R \to R, \cdot : R \to R \to R$ with their standard interpretation. Standard operations can now be expressed:

Name	Symbol	Lambda term
Matrix transposition	$.^T$	$\lambda mij.m_{ji} : M \to M$
Matrix multiplication	$\times_1$	$\lambda mvi.\sum_j m_{ij} \cdot v_j : M \to V \to V$
Cube multiplication	$\times_2$	$\lambda cvij.\sum_k c_{ijk} \cdot v_k : C \to V \to M$
Element wise multiplication	$\odot$	$\lambda uvi.u_i \cdot v_i : V \to V \to V$

5.2.2 Lexical substitution

To obtain a concrete model for a phrase, we need replace the constants c in a proof term by their vectorial representation. This is done by means of a lexicon of semantic terms, that induces a homomorphism $\mathscr{H}$ on terms. Table 1 below gives the substitutions for a contraction-based and an additive-multiplicative model respectively. Translating the abstract proof term from Equation 3 using the contraction-based model means reducing the term below to give the final, classical meaning:

$$(\lambda P.\lambda Q.P \odot Q\ ((\lambda x.x\ (\lambda v.\mathbf{drinks} \times_1 v))\ \mathbf{bob}))((\lambda v.(\mathbf{drinks} \times_1 v))\ \mathbf{alice})$$
$$\to_\beta (\lambda P.\lambda Q.P \odot Q\ ((\lambda v.\mathbf{drinks} \times_1 v)\ \mathbf{bob}))((\lambda v.(\mathbf{drinks} \times_1 v))\ \mathbf{alice})$$
$$\to_\beta (\lambda P.\lambda Q.P \odot Q\ (\mathbf{drinks} \times_1 \mathbf{bob}))(\mathbf{drinks} \times_1 \mathbf{alice})$$
$$\to_\beta (\mathbf{drinks} \times_1 \mathbf{bob}) \odot (\mathbf{drinks} \times_1 \mathbf{alice})$$

As an alternative, we can instantiate the multiplicative-additive model as well, in which the transitive sentences are obtained by adding the individual word embeddings, but the overall result is got by multiplying the two sentence vectors. The final meaning now is

$$(\mathbf{drinks} + \mathbf{alice}) \odot (\mathbf{drinks} + \mathbf{bob})$$

w	$\sigma(w)$	$\mathscr{H}_1(w)$	$\mathscr{H}_2(w)$	$\mathscr{T}(w)$
`cn`	n	$\mathbf{cn}$	$\mathbf{cn}$	V
`adj`	np/n	$\lambda v.(\mathbf{adj} \times_1 v)$	$\lambda v.(\mathbf{adj} + v)$	VV
`adv`	$(np\backslash s)\backslash(np\backslash s)$	$\lambda M.M$	$\lambda M.(\mathbf{adv} + M)$	$(VV)VV$
`itv`	$np\backslash s$	$\lambda v.(\mathbf{itv} \times_1 v)$	$\lambda v.(\mathbf{itv} + v)$	VV
`tv`	$(np\backslash s)/np$	$\lambda uv.(\mathbf{tv} \times_2 v) \times_1 u$	$\lambda uv.(\mathbf{tv} + v + u)$	VVV
`coord`	$(s\backslash s)/s$	$\lambda P.\lambda Q.P \odot Q$	$\lambda P.\lambda Q.(P \odot Q)$	VVV
`quant`	$(s/(np\backslash s))/n$	$\lambda vZ.Z(\mathbf{quant} \times_1 v)$	$\lambda vZ.Z(\mathbf{quant} + v)$	$V(VV)V$

Table 1: Translation that sends abstract terms to a tensor-based model using tensor contraction (column 3) or addition (column 4) as the main operation. In both models the coordinator is interpreted using element wise multiplication. Note that **adj**, **itv**, **tv** and **quant** denote vectors under $\mathscr{H}_2$.

The general description for a simple elliptical phrase that comes out of this is

$$M(\mathbf{alice}, \mathbf{drinks}) \nabla M(\mathbf{bob}, N(\mathbf{drinks}))$$

where M is a general model for an intransitive sentence, and N is a model that could potentially modify the verb tensor. Similarly, the recipe for a transitive elliptical phrase like "Alice drinks beer and Bob does too" will be

$$M(\mathbf{alice}, \mathbf{drinks}, \mathbf{beer}) \nabla M(\mathbf{bob}, N(\mathbf{drinks}), \mathbf{beer})$$

For the sloppy/strict readings involving anaphora, we give the sloppy and strict interpretation in a contraction-based model of the sentence "Gary loves his code and Bill does too":

$$(\mathbf{gary} \times_1 \mathbf{loves} \times_2 (\mathbf{gary} \odot \mathbf{code})) \odot (\mathbf{bob} \times_1 \mathbf{loves} \times_2 (\mathbf{bob} \odot \mathbf{code})) \quad \text{(strict)}$$

$$(\mathbf{gary} \times_1 \mathbf{loves} \times_2 (\mathbf{gary} \odot \mathbf{code})) \odot (\mathbf{gary} \times_1 \mathbf{loves} \times_2 (\mathbf{bob} \odot \mathbf{code})) \quad \text{(sloppy)}$$

So we see that in a classical semantics we can distinguish the two readings and moreover they do not coincide with phrases in which subjects and objects are swapped.

6 Discussion, Conclusion, Further Work

In this paper we incorporated a proper notion of copying into a compositional distributional model of meaning to deal with some selected cases of ellipsis and anaphora. We developed two different concrete vector semantics for an extension of the Lambek Calculus with control modalities that allow for the copying of resources: in the first semantics we followed the categorical framework of [5] and derived sentence meanings similar to the proposal of Kartsaklis [13], but found that for ambiguous cases of ellipsis the meaning of the unambiguous interpretations coincide. In the second semantics, we took a two-step approach by translating proofs to terms of a non-linear lambda terms, that then get concretised in several different models. Here we retain the different semantics for different interpretations of ambiguous elliptical phrases. Some initial experiments have shown, however, that the two frameworks

give comparable results in a similarity task involving ellipsis, suggesting that the entangled semantics serve as a good linear approximation of a non-linear phenomenon.[5]. To complete this work in the future, we would like to carry out a large scale experiment to compare the linear approximative model of the categorical framework, and the classical semantics using non-linear lambda terms. Furthermore, we would like to experiment with the kind of derivational ambiguities that arise in the elliptical setting with anaphora.

Acknowledgements The authors are thankful for the anonymous reviewers' comments. The first author gratefully acknowledges support from a Queen Mary Principal Studentship.

References

[1] Samson Abramsky (2009): *No-cloning in categorical quantum mechanics*. Semantic Techniques in Quantum Computation, pp. 1–28, doi:10.1017/CBO9781139193313.002.

[2] Katalin Bimbó (2014): *Proof theory: Sequent calculi and related formalisms*. CRC Press, doi:10.1201/b17294.

[3] Karel Chvalovskỳ & Rostislav Horčík (2016): *Full Lambek calculus with contraction is undecidable*. The Journal of Symbolic Logic 81(2), pp. 524–540, doi:10.1017/jsl.2015.18.

[4] Stephen Clark (2015): *Vector space models of lexical meaning*. Handbook of Contemporary Semantic Theory, The, pp. 493–522, doi:10.1002/9781118882139.ch16.

[5] Bob Coecke, Edward Grefenstette & Mehrnoosh Sadrzadeh (2013): *Lambek vs. Lambek: Functorial vector space semantics and string diagrams for Lambek calculus*. Annals of pure and applied logic 164(11), pp. 1079–1100, doi:10.1016/j.apal.2013.05.009.

[6] Bob Coecke, Mehrnoosh Sadrzadeh & Stephen Clark (2010): *Mathematical foundations for a compositional distributional model of meaning*. arXiv preprint arXiv:1003.4394. Available at https://arxiv.org/abs/1003.4394.

[7] Mary Dalrymple, Stuart M Shieber & Fernando CN Pereira (1991): *Ellipsis and higher-order unification*. Linguistics and philosophy 14(4), pp. 399–452, doi:10.1007/BF00630923.

[8] Edward Grefenstette & Mehrnoosh Sadrzadeh (2011): *Experimental support for a categorical compositional distributional model of meaning*. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 1394–1404. Available at https://aclanthology.coli.uni-saarland.de/papers/D11-1129/d11-1129.

[9] Bart Jacobs (2011): *Bases as coalgebras*. In: International Conference on Algebra and Coalgebra in Computer Science, Springer, pp. 237–252, doi:10.1007/978-3-642-22944-2_17.

[10] Gerhard Jäger (1998): *A Multi-Modal Analysis of Anaphora and Ellipsis*. University of Pennsylvania Working Papers in Linguistics 5(2), p. 2. Available at https://repository.upenn.edu/pwpl/vol5/iss2/2/.

[11] Gerhard Jäger (2006): *Anaphora and type logical grammar*. 24, Springer Science & Business Media, doi:10.1007/1-4020-3905-0.

[12] Dimitri Kartsaklis (2016): *Coordination in categorical compositional distributional semantics*. arXiv preprint arXiv:1606.01515, doi:10.4204/EPTCS.221.4.

[13] Dimitri Kartsaklis, Matthew Purver & Mehrnoosh Sadrzadeh (2016): *Verb Phrase Ellipsis using Frobenius Algebras in Categorical Compositional Distributional Semantics*. DSALT Workshop, European Summer School on Logic, Language and Information. Available at https://www.eecs.qmul.ac.uk/~mpurver/papers/kartsaklis-et-al16dsalt.pdf.

[5] This work is currently under review and could therefore not be included as of yet.

[14] Dimitri Kartsaklis & Mehrnoosh Sadrzadeh (2013): *Prior disambiguation of word tensors for constructing sentence vectors*. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1590–1601. Available at https://aclanthology.coli.uni-saarland.de/papers/D13-1166/d13-1166.

[15] Ruth Kempson, Ronnie Cann, Eleni Gregoromichelaki Arasheshghi & Matthew Purver (2015): *Ellipsis*. *Chapter 4 of The Handbook of Contemporary Semantic Theory* 3, p. 114, doi:10.1002/9781118882139.ch4.

[16] Dekang Lin & Patrick Pantel (2001): *DIRT@ SBT@ discovery of inference rules from text*. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 323–328, doi:10.1145/502512.502559.

[17] Michael Moortgat (1996): *Multimodal linguistic inference*. *Journal of Logic, Language and Information* 5(3-4), pp. 349–385, doi:10.1007/BF00159344.

[18] Michael Moortgat & Gijs Wijnholds (2017): *Lexical and Derivational Meaning in Vector-Based Models of Relativisation*. *Proceedings of the 21st Amsterdam Colloquium*. Available at https://arxiv.org/abs/1711.11513.

[19] Glyn Morrill & Oriol Valentín (2015): *Computational Coverage of TLG: Nonlinearity*. In: *Proceedings of NLCS'15. Third Workshop on Natural Language and Computer Science*, 32, EasyChair Publications, pp. 51–63. Available at https://arxiv.org/abs/1706.03033.

[20] Glyn Morrill & Oriol Valentín (2016): *On the Logic of Expansion in Natural Language*. In: *Logical Aspects of Computational Linguistics. Celebrating 20 Years of LACL (1996–2016) 9th International Conference, LACL 2016, Nancy, France, December 5-7, 2016, Proceedings 9*, Springer, pp. 228–246, doi:10.1007/978-3-662-53826-5_14.

[21] Reinhard Muskens & Mehrnoosh Sadrzadeh (2016): *Context Update for Lambdas and Vectors*. In: *International Conference on Logical Aspects of Computational Linguistics*, Springer, pp. 247–254, doi:10.1007/978-3-662-53826-5_15.

[22] Mehrnoosh Sadrzadeh, Stephen Clark & Bob Coecke (2013): *The Frobenius anatomy of word meanings I: subject and object relative pronouns*. *Journal of Logic and Computation* 23(6), pp. 1293–1317, doi:10.1093/logcom/ext044.

[23] Bayu Surarso & Horoakira Ono (1996): *Cut elimination in noncommutative substructural logics*. *Reports on Mathematical Logic* 30, pp. 13–29. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.95.5770&rep=rep1&type=pdf.

[24] Peter D Turney & Patrick Pantel (2010): *From frequency to meaning: Vector space models of semantics*. *Journal of artificial intelligence research* 37, pp. 141–188, doi:10.1613/jair.2934.

[25] Gijs Wijnholds (2014): *Categorical foundations for extended compositional distributional models of meaning*. *MSc. thesis*. Available at https://pdfs.semanticscholar.org/19c9/5240e4a37603a74e76831f0c2e4673de40fc.pdf.

A Proofs

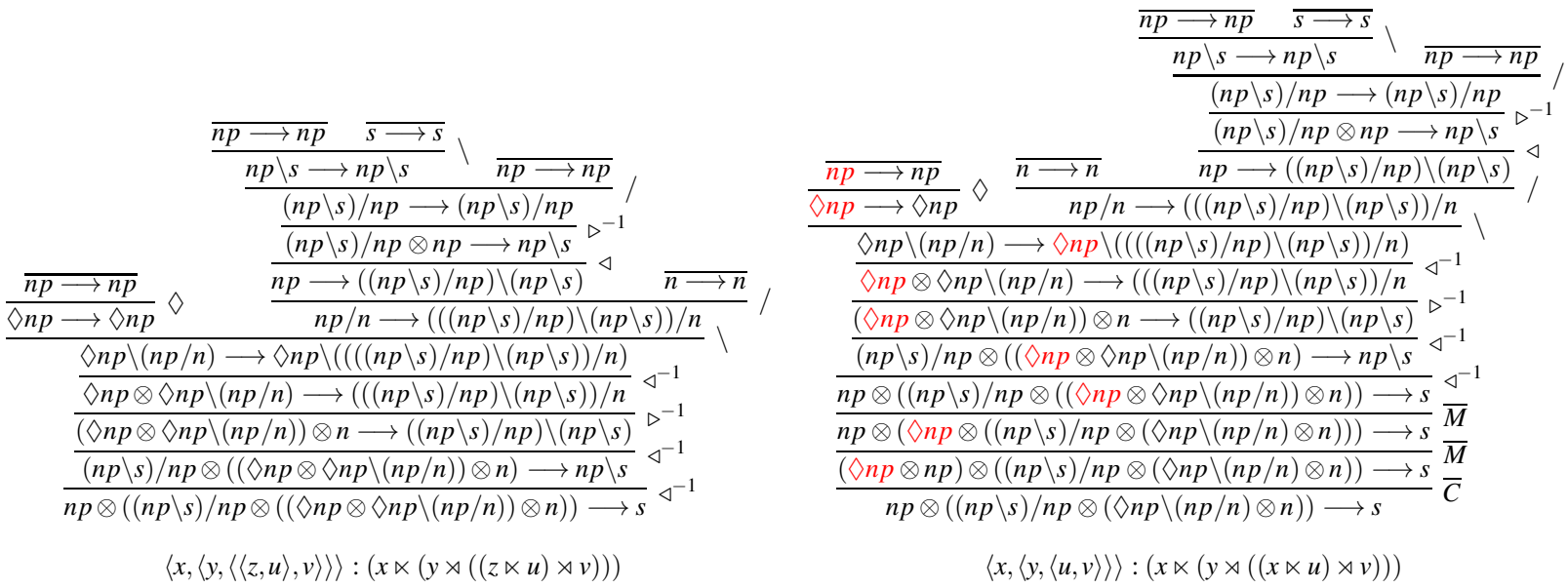


Figure 4: Derivation of "x loves Gary his code" (left) and "x loves his code" (right)

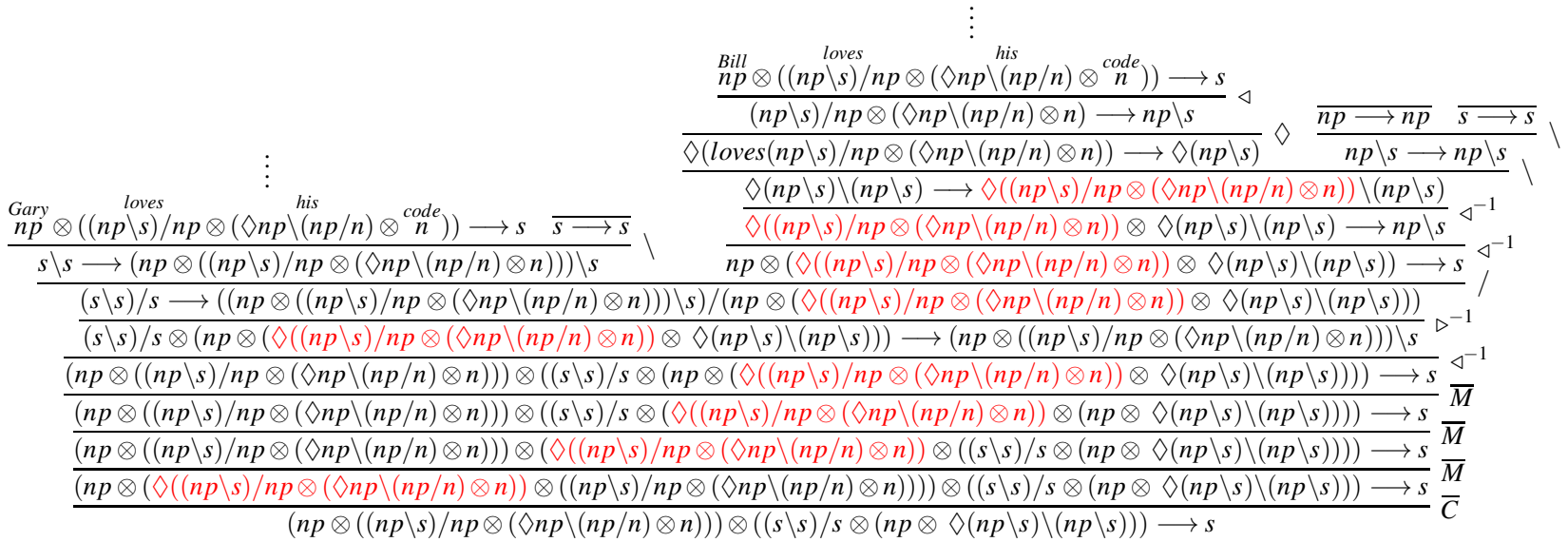


Figure 5: Sloppy reading of gary's code. "loves his code" is copied to the two main subproofs, where each one is resolved with their respective noun phrase argument (left: Gary, right: Bill).

$$\vdots$$

$$\overset{\textit{Bill}}{np} \otimes ((\overset{\textit{loves}}{np\backslash s)/np} \otimes ((\overset{\textit{Gary}}{\Diamond np} \otimes \overset{\textit{his}}{\Diamond np\backslash(np/n)}) \otimes \overset{\textit{code}}{n})) \longrightarrow s \quad \triangleleft$$

$$(np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n) \longrightarrow np\backslash s \quad \Diamond$$

$$\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \longrightarrow \Diamond(np\backslash s)$$

$$\dfrac{np \longrightarrow np \qquad s \longrightarrow s}{np\backslash s \longrightarrow np\backslash s} \ \backslash$$

$$\Diamond(np\backslash s)\backslash(np\backslash s) \longrightarrow \Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n))\backslash(np\backslash s) \quad \backslash$$

$$\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes \Diamond(np\backslash s)\backslash(np\backslash s) \longrightarrow np\backslash s \quad \triangleleft^{-1}$$

$$np \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes \Diamond(np\backslash s)\backslash(np\backslash s)) \longrightarrow s \quad \triangleleft^{-1}$$

$$\vdots$$

$$\overset{\textit{Gary}}{np} \otimes ((\overset{\textit{loves}}{np\backslash s)/np} \otimes ((\overset{\textit{Gary}}{\Diamond np} \otimes \overset{\textit{his}}{\Diamond np\backslash(np/n)}) \otimes \overset{\textit{code}}{n})) \longrightarrow s \qquad s \longrightarrow s \quad \backslash$$

$$s\backslash s \longrightarrow (np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)))\backslash s \quad /$$

$$(s\backslash s)/s \longrightarrow ((np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)))\backslash s)/(np \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \quad \triangleright^{-1}$$

$$(s\backslash s)/s \otimes (np \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow (np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)))\backslash s \quad \triangleleft^{-1}$$

$$(np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n))) \otimes ((s\backslash s)/s \otimes (np \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes \Diamond(np\backslash s)\backslash(np\backslash s)))) \longrightarrow s \quad \overline{M}$$

$$(np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n))) \otimes ((s\backslash s)/s \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s)))) \longrightarrow s \quad \overline{M}$$

$$(np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n))) \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s)))) \longrightarrow s \quad \overline{M}$$

$$(np \otimes (\Diamond((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)) \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n)))) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow s \quad \overline{C}$$

$$(np \otimes ((np\backslash s)/np \otimes ((\Diamond np \otimes \Diamond np\backslash(np/n)) \otimes n))) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow s \quad \overline{M}$$

$$(np \otimes (\Diamond np \otimes ((np\backslash s)/np \otimes (\Diamond np\backslash(np/n) \otimes n)))) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow s \quad \overline{M}$$

$$((\Diamond np \otimes np) \otimes ((np\backslash s)/np \otimes (\Diamond np\backslash(np/n) \otimes n))) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow s \quad \overline{C}$$

$$(np \otimes ((np\backslash s)/np \otimes (\Diamond np\backslash(np/n) \otimes n))) \otimes ((s\backslash s)/s \otimes (np \otimes \Diamond(np\backslash s)\backslash(np\backslash s))) \longrightarrow s$$

Figure 6: Strict reading of gary's code. First, "Gary" is copied and resolved with "loves his code", after which "loves Gary his code" is copied to the two main subproofs (left: Gary, right: Gary).