Compositional Distributional Cognition

Yaared Al-Mehairi

with Bob Coecke

Department of Computer Science

University of Oxford

{*khalid.al-mehairi, coecke*}*@cs.ox.ac.uk*

A thesis submitted for the degree of

Master of Science in Computer Science

Trinity 2015

Abstract

Building on existing compositional distributional models of meaning [12, 47], we present an adaptation of the Integrated Connectionist/Symbolic Architecture (ICS) *introduced in [52]. In particular, we build a novel compositional distributional model of cognition (*DisCog*) that resolves representational issues in* ICS *via mapping of same-type symbolic structures to a finite shared meaning space. In addition, we further improve on* ICS *by means of an important extension to genuine meaning vectors. What's more, we give a functional account of harmony in* DisCog *derived from our categorical setting.*

Clear eyes. Full hearts. Can’t lose.

Acknowledgements

Professor Bob Coecke and Martha Lewis.

Garry Shandling, Jerry Seinfeld, Larry David, and Johnny Knoxville.

Tucson, Addis Ababa, London, Princeton, San Francisco, and Oxford.

Mom and Dad.

Contents

List of Figures

List of Tables

List of Algorithms

The mind is like an iceberg, it floats with one-seventh of its bulk above water.

— Sigmund Freud

1 Introduction

Attempts to understand the mind and its operation date back to Ancient Times. According to Plato, the mind is independent of the body and belongs to the world of Forms[1]. Plato's belief that the mind is separate from the body provides an explanation for the nature of human knowledge: in view of being *aspatial* and *atemporal*, the mind can experience true reality, and therefore access universal truths [45]. For Aristotle, the mind is a property of the body, but intellect is unique, i.e. aspatial, because the bounds of human consciousness are not restricted in the same sense that human faculties of perception are [2]. Since the early days of philosophical thought, thinkers have failed to agree on a theory of the mind, and the debate continues today. At present, the study of the mind has become the purview of not only philosophers, but also psychologists, neuroscientists, anthropologists, linguists, and computer scientists. Cognitive science is the interdisciplinary scientific investigation of the mind and its processes. The central tenet of cognitive science is that 'thinking' can best be understood in terms of representational structures in the mind and computational procedures that operate on those structures. There is much dispute regarding the nature of the representations and computations that constitute 'thinking'. That being said, most work in cognitive science assumes that representations in the mind are analogous to computer data structures, and computational procedures in the mind are similar to computational algorithms. As such, cognitive science boils down to the task of answering one question: What type of computer is the human mind/brain?

[1] Forms are non-material abstract ideas, existing neither in space nor time, that realize genuine knowledge.

1.1 Background

In the cognitive science community, there are two competing approaches to the computational modelling of the mind: *connectionist* and *combinatorial*. The connectionist, i.e. *distributional*, approach models the mind/brain as a massively parallel computer consisting of billions of basic processing units, i.e. neurons. Each unit has an activation level and is linked to other units by way of weighted connections. The activation value of an unit is a simple function of the activation values of neighboring units and the corresponding connection weights. Learning is the determination of correct weights via statistical analysis of experience. The framework used to manipulate these structures is the mathematics of vector and matrix algebra. The combinatorial, i.e. *symbolic* or *compositional*, strategy is to treat the mind/brain as a serial device that manipulates discrete structures consisting of abstract symbols to compute recursive functions.[2] A fundamental theoretical question in cognitive science is: Can the connectionist and combinatorial approaches be integrated in a useful and meaningful fashion?

Traditional thought supports the position that the connectionist and combinatorial approaches are orthogonal. While the former presents the mind/brain as a noisy, massively parallel numerical computer, the latter depicts the mind/brain as a well-organized, rule-governed processor of discrete symbolic structures. However, recent developments in cognitive science suggest that there is some middle ground on which we can view the mind/brain as a matter of both parallel distributed processing architecture and symbolic architecture.

[52] proposes a novel solution to unify the connectionist and symbolic architectures. The new architecture is called the *Integrated Connectionist/Symbolic Architecture* (ICS). What distinguishes ICS from other computational models of mind is an account of two methods of decomposing representations corresponding to two different levels of structure: one is *functionally* relevant, the other *process*-relevant. In this *split-level* architecture, connectionist and symbolic computational descriptions each play an essential role in overall cognitive explanation.

At the functional level, the relevant decomposition of representations is into constituents, just as in symbolic theory. Representations are associated in the way the combinatorial strategy requires, and this is what provides the explanation of cognitive productivity[3]. The decomposition of representations into constituents is

[2] The principle of compositionality states that the meaning of a complex expression is determined by the meaning of its parts and the rules used for combining them.

[3] Cognitive productivity refers to our ability to process an unbounded number of distinct inputs.

also what specifies the *meaning* of representations.

However, at the process level, the relationships among representations are not established in a serial, constituent-by-constituent process, as required by the combinatorial strategy. There is no step-by-step algorithm defined over constituents that describes the moment-by-moment mechanisms underlying cognition. The latter requires algorithms that are defined via the process-relevant decomposition which decomposes a representation not into its constituents, but into its *activation values*. The dynamics of cognition must be characterized by connectionist algorithms.

What makes ICS possible is the discovery that there is a formal equivalence or *isomorphism* between constituent decomposition in symbolic representations and in vectorial representations. This isomorphism is codified in *tensor product representations*. In ICS, a crucial transformation occurs between higher and lower sublevels of the computational level: the higher sublevel aligns with symbolic constituents, the lower sublevel aligns with individual neurons. ICS provides a fully formal reduction, as required by a computational theory: a formal mapping from symbolic structures to activity vectors, another formal mapping from activity vectors to individual unit activity values, and finally (in principle) a formal mapping from unit activities to neural activities. Hence, ICS reduces abstract cognitive functions to elementary operations that fall within the computational capabilities of neural networks, i.e. linear associators:

$$\mathbf{o} = \mathbb{W} \cdot \mathbf{i} \tag{1.1}$$

An understanding of how cognition can arise in a brain is of great philosophical and practical importance. Knowledge of the primitive operations made available by a brain, to which complex cognitive functions must be reduced by computational theories, constitutes a significant advancement in cognitive science that is of interest to areas of application in artificial intelligence, such as natural language processing (NLP), computer vision, and robotics, to name but a few.

1.2 Motivation

The design of ICS is a promising and exciting development in the computational modelling of the mind. The tensor product representations used in [52] for combining connectionist and symbolic representations extend to any structure that can be represented in terms of filler/role bindings (more on this later). However, the

tensor product representations do admit some weaknesses. Firstly, the representational space for a concept *grows* in size as more elements are added to the compound. This means that compounds can become unwieldy since concepts exist in an unbounded representational space. Secondly, it is only possible to compare symbolic representations of the *same* type, where type here refers to the tuple of compositional relations obtained by traversing the symbolic structure in some canonical order. This means that in a NLP task, for instance, the symbolic representations (e.g. a parse tree, a dependency graph, a set of predicate argument relations, etc.) of *Comedians tell jokes* and *Comedians tell funny jokes* cannot be compared at the sentence level.

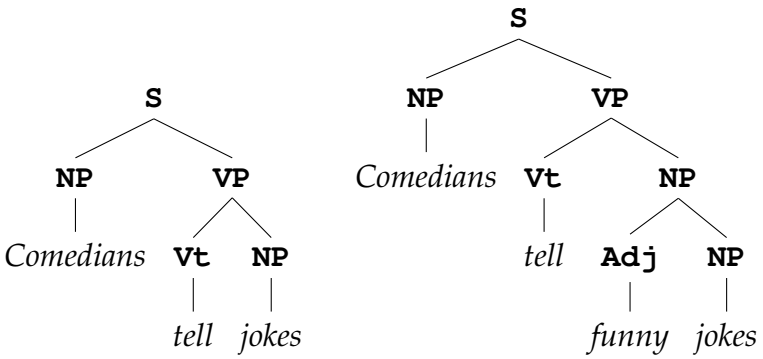


Figure 1.1: Example parse trees

[8] introduces the compositional distributional (DISCO) model of semantics, which represents the meaning of words as vectors and uses the tensor product to combine words and their grammatical roles in sentences. The suggestion is that the limitations of the tensor product representations can be overcome by using a more abstract representation. The mathematical theory of quantum mechanics (QM) is based on Hilbert spaces, and composite systems in QM, formed by interacting quantum-mechanical systems, are realized using tensor products. The objects in which [8] situates representations is Hilbert spaces. Moreover, the operator proposed for combining representations is the tensor product, as suggested in [52]. This link indicates that ICS which uses vector spaces may benefit from borrowing more from the well-developed mathematical theory of QM.

QM inspired models of meaning view concepts as subspaces of a vector space. Similarity of one concept to another is measured by the projection of one concept onto another. [7, 12] develop the DISCO model by utilising grammar in order to use composite spaces without increase in size of the resulting meaning space. The

meaning of a composite concept arises from the meaning of its constituents mediated by grammar, i.e. by accounting for the interaction of its constituents via their relational roles. This allows composite concepts to be directly compared with their constituents, and the meaning of sentences of varying length and type to be compared. What's more, the DISCO model of semantics provides natural explanations for qualitative phenomena, such as the 'Pet Fish' phenomenon [36], which constitutes important progress in characterizing human concept use.

The DISCO account of semantics unifies compositional descriptions of semantics, where the meaning of a sentence is seen as a function of the meanings of the individual words of the sentence[4], and distributional descriptions of semantics, where the meaning of individual words are characterized as vectors[5].

Syntactic Analysis	Semantic Interpretation
S $\rightarrow$ **NP VP**	**VP (NP)**
NP $\rightarrow$ **Adj NP**	**Adj (NP)**
NP $\rightarrow$ *comedians, jokes, book, etc.*	*comedians, etc.*
Adj $\rightarrow$ *funny, loud, bright, etc.*	$\lambda x.funny(x)$, *etc.*
VP $\rightarrow$ **Vt NP**	**Vt (NP)**
Vt $\rightarrow$ *tell, bring, make, etc.*	$\lambda y.\lambda x.tell(x,y)$, *etc.*

Table 1.1: A simple compositional semantic model

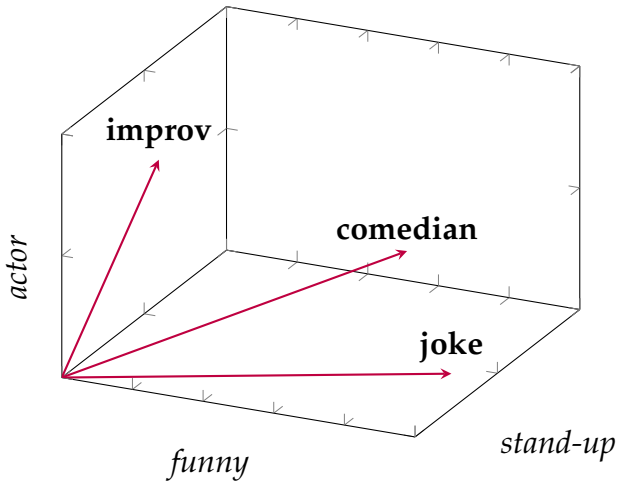


Figure 1.2: A simple distributional semantic model

[4]Montague's central assertion captures the spirit of the model-theoretic approach to semantics: "I reject the contention that an important theoretical difference exists between formal and natural languages" [40].

[5]Firth's famous dictum neatly expresses the idea behind the distributional hypothesis: "You shall know a word by the company it keeps" [15].

The compositional and distributional descriptions are linked by the fact that they share the common structure of a *compact closed category*. This enables the compositional rules of the (pregroup) grammar to be applied in the vector space model to map syntactically well-formed strings of words into one shared meaning space. The mathematical structure employed admits a purely diagrammatic calculus (i.e. string diagrams) which exposes information flow between constituents.

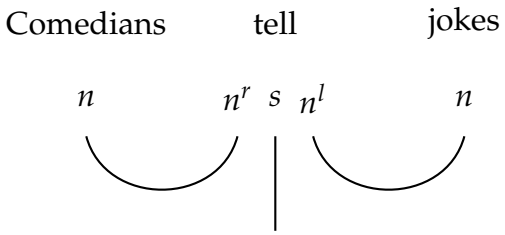


Figure 1.3: Example pregroup parse

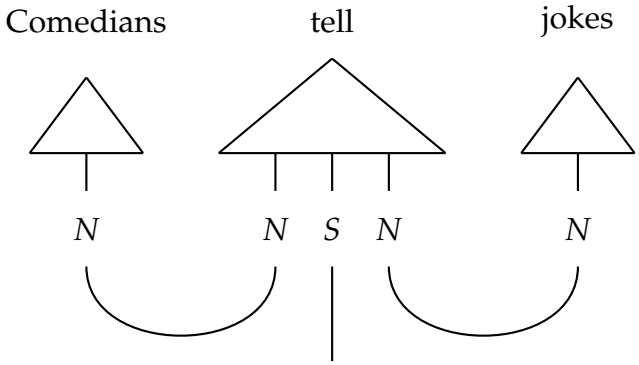


Figure 1.4: Example string diagram

Our proposal is that the interaction of QM and ICS is a fruitful area of research for cognitive science. The power of ICS derives from the novel aspects of the architecture: connectionist computation is made more powerful in order to meet the strong constraint that it serve as the lower-level platform for what emerges as symbolic computation at a higher level of description, and symbolic theory is enriched because the symbolic level emerges from a lower-level connectionist substrate. Of particular interest are connectionist computational properties which percolate up to the symbolic level, enriching it with novel concepts, much like the manner by which, in physics, gas – a continuous, extended system defined by properties like volume, pressure, temperature, and entropy – is a higher-level description of a system that on a lower level is described as a collection of discrete, point-like atoms

individually possessing none of these properties but others instead, such as velocity. The central such property addressed in [52] is *optimization*. Connectionist computation often computes representations that optimize well-formedness or *harmony*. Optimization turns out to provide many novel conceptual and technical tools for formally characterizing the nature of knowledge. We set out to build a concrete QM model of ICS based on the DISCO framework (DISCOG) and recover this fundamental notion of harmony.

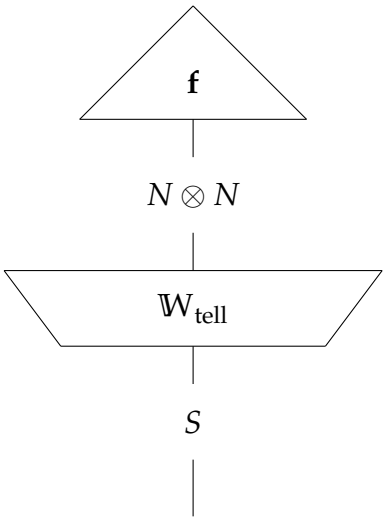


Figure 1.5: Example filler/role binding in DISCOG

1.3 Scope and Contribution

In this thesis, we present an adaptation of ICS derived from the DISCO model of semantics. Our new model of cognition, DISCOG, solves the representational problems of ICS via mapping of same-type symbolic structures to a *finite shared* meaning space. Furthermore, in that DISCOG works with authentic meaning vectors in contrast to generic ones as in ICS, our model improves on ICS by means of an important extension to *genuine* meaning representations. We achieve these results by reformulating the DISCO model of semantics as the recursive realization of filler/role bindings which allows us to give a connectionist account of compositional distributional information flow. In addition, we provide simple algorithms to construct linear associators characterizing these primitive compositional operations. We then discuss the problem of unbinding (fillers from roles) and put forward a provisional solution based on the Moore-Penrose pseudo inverse. This

allows us to demonstrate how massively parallel distributed processing at the connectionist level realizes arbitrarily complex (recursive) functions at the symbolic level. Lastly, we give a *functional* account of harmony in DISCOG extracted from our categorical setting.

1.4 Structure

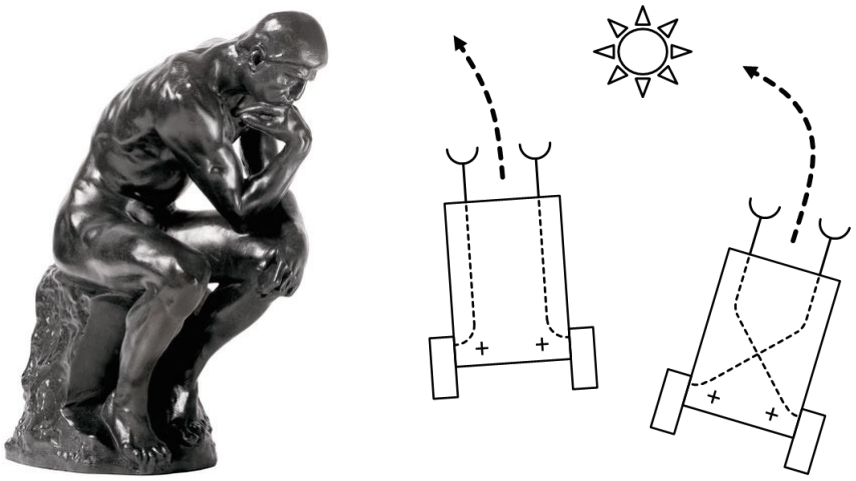
Here, we outline the structure of this work. We begin with an extensive presentation of the ICS model introduced in [52] (**Harmonic Mind**). Next, we give an overview of the DISCO model of semantics [12] and a relevant extension concerning Frobenius algebra [47] (**Compositional Distributional Model of Semantics**). We then present our *entirely* new model of cognition, DISCOG (**Compositional Distributional Cognition**). We end with a brief recap of our results and some suggestions for further investigations (**Conclusion and Future Work**).

When you come to a fork in the road, take it.
— Yogi Berra

2 Harmonic Mind

What type of computer is the human mind/brain? The answer [52] proposes is: The mind is a symbol-manipulating computer, an abstract virtual machine realized in a brain performing connectionist computation. This characterizes the core of *higher* cognition, the particularly challenging and important realm of cognitive faculties operating in domains like abstract reasoning, planning, and language – faculties that are highly developed primarily in the *human* mind/brain. Generally, cognitive phenomena are understood at various levels of description, i.e. high- and low-level cognitive processes. Whereas high-level cognition is mostly ascribed to human cognitive processing, low-level cognition, such as sensory processing, exists in virtually all animal species. The difference between high- and low-level cognition can be symbolized by the contrast between Auguste Rodin's "The Thinker" (Figure 2.1a) and a Braitenberg vehicle (Figure 2.1b). "The Thinker" being deep in thought while completely immobilized embodies the decoupling of high-level cognitive processing and actions in the world. Contrary to this, a Braitenberg vehicle represents a direct interaction of sensory processing and corresponding actions, and is therefore an example of pure low-level cognition [31].

Here, we introduce the formal foundations of the *Integrated Connectionist/Symbolic Architecture* (ICS) presented in [52]. ICS is defined by general principles that relate the aggregate properties of connectionist representations and processes with symbolic representations and processes. These principles identify optimization as a central organizing principle that spans the levels of both the connectionist and the symbolic descriptions. This work provides the first abutment of the bridge – linking cognitive science and quantum mechanics – which we ultimately aim to build.



(a) The Thinker *(b) Braitenberg vehicle*

Figure 2.1: Contrasting high- and low-level cognition

2.1 Representation

Information is represented in the mind/brain by massively distributed activity patterns that, for prominent aspects of higher cognition, possess global structure describable through symbolic discrete data structures. The *isomorphism* between activation values of connectionist units and symbolic mental representations is codified in *tensor product representations*.

2.1.1 Linear Approximation

The most basic question about representation concerns realizations of composite structures in distributed activation vectors.[1] The value of an operation used to combine elements rests in its compatibility with developing connectionist structuring operations. In this regard, with respect to central phenomena of higher cognition, the *linear approximation* suffices for the analysis of the network realization of symbolic representations. However, future work on ICS theory will no doubt pursue nonlinear extensions of the methods developed up until now.

Definition 2.1 (Superposition Principle)**.** *The realization of the set* $\{\mathtt{A}, \mathtt{B}\}$ *is the* ***superposition*** *or* ***vector sum*** *of the activation vectors realizing* $\mathtt{A}$ *and* $\mathtt{B}$*.*

[1] Jerome Feldman's two-horse problem colorfully poses the issue underlying the realization of composite structures: "If the representation of *one* horse fills up an entire network, how is it possible to represent *two* horses?"[14].

2.1.2 Inverting Superposition

The Superposition Principle presents the following question: In what sense does a single composite pattern represent multiple elements? Representation in ICS ensures that given an activity pattern realizing a set of symbols, there is one and only one set of symbols realized by that pattern. Hence, an ICS network processes composite representations in a way that is sensitive to the presence of individual parts.

Definition 2.2 (Linear Independence)**.** *A subset of vectors* $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ *is linearly independent if no non-trivial linear combination of them vanishes, i.e.*

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_n\mathbf{v}_n = \mathbf{0} \tag{2.1}$$

only if $c_1 = c_2 = \cdots = c_n = 0$.

Definition 2.3 (Independence Assumption)**.** *In a connectionist realization of symbolic computation, the activation vectors realizing the atomic symbols, and those realizing the role vectors, are (**linearly**) **independent**.*

2.1.3 Filler/Role Binding

A set is an unstructured collection of elements. Thus, for example, $\{\mathtt{A}, \mathtt{B}\}$ and $\{\mathtt{B}, \mathtt{A}\}$ denote the same set. When we move to true symbol *structures*, we must start to distinguish among the different *roles* in the overall structure played by different symbol tokens. To overcome the $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$ problem (i.e. the commutative property of vector addition), symbol tokens are bound to variable roles to individuate a particular structure within a general class. The class of a structure is determined by its roles (e.g. the class of strings is determined by positional roles).

Definition 2.4 (Filler/Role Decomposition)**.** *A structure is a set of **bindings** of various structural **roles** to their **fillers**.*

The internal structure of a vector that realizes the binding of a symbol token to a role is the *tensor product* of the filler and role vectors.

Definition 2.5 (Tensor Product of Vectors)**.** *Let* $\mathbf{v}, \mathbf{w}$ *be vectors of dimension* m *and* n*, respectively. Their tensor product* $\mathbf{v} \otimes \mathbf{w}$ *is the* mn*-dimensional vector with components*

$$[\mathbf{v} \otimes \mathbf{w}]_{ij} = [\mathbf{v}]_i [\mathbf{w}]_j \tag{2.2}$$

where

$$[\mathbf{v}]_i \tag{2.3}$$

is the i^{th} *component of* $\mathbf{v}$.

Definition 2.6 (Tensor Product Binding)**.** *The binding* $\mathtt{f}/r$ *of a filler* $\mathtt{f}$ *to a role* r *is realized as a vector*

$$\mathbf{f}/\mathbf{r} = \mathbf{f} \otimes \mathbf{r} \tag{2.4}$$

that is the tensor product of a vector $\mathbf{f}$ *realizing* $\mathtt{f}$ *with a vector* $\mathbf{r}$ *realizing* r.

Claim 2.1 (Binding Independence)**.** *Let the independence assumption be satisfied. Then it follows that the binding vectors* $\mathbf{f} \otimes \mathbf{r}$ *– as* $\mathbf{f}$ *ranges over the realizations of all the atomic symbols, and* $\mathbf{r}$ *over the realizations of all the roles – will form a linearly independent set.*

Proof. The tensor product of vectors preserves their linear independence. If the vectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ and $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ are, respectively, linearly independent, then the vectors $\{\mathbf{v}_i \otimes \mathbf{w}_j\}_{ij}$ are also linearly independent.

□

2.1.4 Recursive Connectionist Realization

Embedding and recursion requires the tensor product representation to handle embedded structure, where the filler is itself a complex structure, and not an atomic symbol. The solution presented is to recursively use tensor product realization for the complex fillers (or roles). The case of binary trees is used for relevance to linguistics and computer data structures.

Definition 2.7 (Connectionist Realization)**.** *A symbolic structure* $\mathtt{s}$ *is defined by a collection of structure roles* $\{r_i\}$ *each of which may be occupied by a filler* $\mathtt{f}_i$. $\mathtt{s}$ *is a set of constituents, each a filler/role binding* $\mathtt{f}_i/r_i$. *The connectionist realization of* $\mathtt{s}$ *is an activation vector*

$$\mathbf{s} = \sum_i \mathbf{f}_i \otimes \mathbf{r}_i \tag{2.5}$$

that is the sum of vectors realizing the filler/role bindings.

Let **s = [NP VP]** be a binary tree with left and right subtrees **NP** and **VP**:

Figure 2.2: Example binary trees

Let $\mathbf{s}, \mathbf{v}_1, \mathbf{v}_2$ be the vectors realizing the trees **s**, **NP**, **VP**. The connectionist realization of **s** is:

$$\mathbf{s} = \mathbf{v}_1 \otimes \mathbf{r}_0 + \mathbf{v}_2 \otimes \mathbf{r}_1 \tag{2.6}$$

If **VP** is a tree rather than an atomic symbol, it can be expressed in terms of its left and right subtrees **VP = [Vt NP]**. Let $\mathbf{v}_3, \mathbf{v}_4$ be the vectors realizing the trees **Vt**, **NP**:

$$\mathbf{v}_2 = \mathbf{v}_3 \otimes \mathbf{r}_0 + \mathbf{v}_4 \otimes \mathbf{r}_1 \tag{2.7}$$

Therefore, the structure **s = [NP [Vt NP]]**:

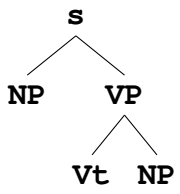


Figure 2.3: Graphical representation of **s**

has the following connectionist representation:

$$\begin{aligned}
\mathbf{s} &= \mathbf{v}_1 \otimes \mathbf{r}_0 + \mathbf{v}_2 \otimes \mathbf{r}_1 && (2.8)\\
&= \mathbf{v}_1 \otimes \mathbf{r}_0 + (\mathbf{v}_3 \otimes \mathbf{r}_0 + \mathbf{v}_4 \otimes \mathbf{r}_1) \otimes \mathbf{r}_1 && (2.9)\\
&= \mathbf{v}_1 \otimes \mathbf{r}_0 + (\mathbf{v}_3 \otimes \mathbf{r}_0) \otimes \mathbf{r}_1 + (\mathbf{v}_4 \otimes \mathbf{r}_1) \otimes \mathbf{r}_1 && (2.10)\\
&= \mathbf{v}_1 \otimes \mathbf{r}_0 + \mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1) + \mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1) && (2.11)\\
&\equiv \mathbf{v}_1 \otimes \mathbf{r}_0 + \mathbf{v}_3 \otimes \mathbf{r}_{01} + \mathbf{v}_4 \otimes \mathbf{r}_{11} && (2.12)
\end{aligned}$$

Representations (2.8) and (2.12) contrast the complex filler and complex role perspectives. This expansion of vectors realizing trees into the vectors realizing their subtrees may be recursively continued until the atomic symbols at the leaves of the tree have been reached.

Definition 2.8 (Recursive Role Vectors for Binary Trees)**.** *Let x be a bit string, where $|x|$ denotes the number of bits ($|\varepsilon| = 0$). Then*

$$\mathbf{r}_{left-child(x)} = \mathbf{r}_{0x} = \mathbf{r}_0 \otimes \mathbf{r}_x \qquad \mathbf{r}_{right-child(x)} = \mathbf{r}_{1x} = \mathbf{r}_1 \otimes \mathbf{r}_x \tag{2.13}$$

Regardless of the representation (i.e. complex filler or complex role), it is unclear in what vector space this addition is performed. Symbols at depth d in a binary tree are realized by $\mathcal{S}_{(d)}$, the FR^d-dimensional vector space formed from vectors of the form $\mathbf{f} \otimes \mathbf{r}_i \otimes \mathbf{r}_j \otimes \cdots \otimes \mathbf{r}_k$ with d role vectors. A vector space containing all the vectors in $\mathcal{S}_{(d)}$ for all d is the direct sum:

$$\mathcal{S}^* \equiv \mathcal{S}_{(0)} \oplus \mathcal{S}_{(1)} \oplus \mathcal{S}_{(2)} \oplus \cdots \tag{2.14}$$

A vector $\mathbf{s}$ in this space is a sequence of vectors $(\mathbf{s}_{(0)}; \mathbf{s}_{(1)}; \mathbf{s}_{(2)}; \cdots)$ where $\mathbf{s}_{(d)}$ is a vector in $\mathcal{S}_{(d)}$. $\mathbf{s}$ is also written $\mathbf{s}_{(0)} \oplus \mathbf{s}_{(1)} \oplus \mathbf{s}_{(2)} \oplus \cdots$. So (2.11) can be interpreted as:

$$\mathbf{s} = [\mathbf{v}_1 \otimes \mathbf{r}_0] \oplus [\mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1) + \mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1)] \tag{2.15}$$

Alternatively, it is natural to view each $\mathbf{s}_{(d)}$ as embedded in $\mathcal{S}^*$. Calling the embedded vector '$\mathbf{s}_{(d)}$':

$$\text{`}\mathbf{s}_{(d)}\text{'} \equiv \mathbf{0}_{(0)} \oplus \mathbf{0}_{(1)} \oplus \cdots \oplus \mathbf{0}_{(d-1)} \oplus \mathbf{s}_{(d)} \oplus \mathbf{0}_{(d+1)} \oplus \cdots \tag{2.16}$$

where $\mathbf{0}_{(k)}$ is the zero vector of $\mathcal{S}_{(k)}$. That is, '$\mathbf{s}_{(d)}$' is the element of $\mathcal{S}^*$ with $\mathbf{s}_{(d)}$ in the sequence position appropriate for its space $\mathcal{S}_{(d)}$, and zero everywhere else. So (2.11) can also be interpreted as:

$$\mathbf{s} = \text{`}[\mathbf{v}_1 \otimes \mathbf{r}_0]\text{'} \oplus \text{`}[\mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1) + \mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1)]\text{'} \tag{2.17}$$

Writing this equation out explicitly in terms of sequences of vectors gives (2.18):

$$\begin{array}{rcccccc}
 & (\mathbf{s}_{(0)}; & \mathbf{s}_{(1)}; & & \mathbf{s}_{(2)}; & \mathbf{s}_{(3)}; & \cdots) \\
\text{`}\mathbf{v}_1 \otimes \mathbf{r}_0\text{'} = & (\mathbf{0}_{(0)}; & \mathbf{v}_1 \otimes \mathbf{r}_0; & & \mathbf{0}_{(2)}; & \mathbf{0}_{(3)}; & \cdots) \\
+\ \text{`}\mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1)\text{'} = & (\mathbf{0}_{(0)}; & \mathbf{0}_{(1)}; & & \mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1); & \mathbf{0}_{(3)}; & \cdots) \\
+\ \text{`}\mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1)\text{'} = & (\mathbf{0}_{(0)}; & \mathbf{0}_{(1)}; & & \mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1); & \mathbf{0}_{(3)}; & \cdots) \\
\hline
\mathbf{s} = & (\mathbf{0}_{(0)}; & \mathbf{v}_1 \otimes \mathbf{r}_0; & \mathbf{v}_3 \otimes (\mathbf{r}_0 \otimes \mathbf{r}_1) + \mathbf{v}_4 \otimes (\mathbf{r}_1 \otimes \mathbf{r}_1); & & \mathbf{0}_{(3)}; & \cdots)
\end{array} \tag{2.18}$$

Thus interpreted, the only type of vector sum to employ is the ordinary operation: add corresponding numerical components.

2.2 Processing

Information is processed in the mind/brain by widely distributed connection patterns (i.e. weight matrices) that, for central aspects of higher cognition, possess global structure describable through symbolic expressions for recursive functions.

2.2.1 Linear Processing

Activation vectors encode information in connectionist networks. Information processing in parallel distributed processing (PDP) networks is the spread of activation. This process utilizes the central operation of vector space theory – matrix-vector multiplication – by applying the matrix of connection weights to the vector of activations. In its purest form, PDP simply *is* matrix multiplication.

Definition 2.9 (Linear Associator)**.** *The core connectionist processing operation is the multiplication of the input activity vector* $\mathbf{i}$ *by the matrix of connection strengths* $\mathbb{W}$

$$\mathbf{o} = \mathbb{W} \cdot \mathbf{i} \tag{2.19}$$

where ' $\cdot$ *' denotes matrix-vector multiplication and*

$$o_\beta = \sum_\alpha W_{\beta\alpha} i_\alpha \equiv W_{\beta 1} i_1 + W_{\beta 2} i_2 + \cdots \tag{2.20}$$

where $\mathbb{W}_{\beta\alpha}$ *is the weight of the connection to output unit* β *from input unit* α*.*

2.2.2 Symbolic Computation

The mathematical isomorphism between symbolic structure and vector structure that is exploited by the tensor product representations extends to the *basic* operations of symbolic computation. These operations can be realized as the simplest type of connectionist operation, multiplication by a weight matrix. That is, they can be computed by the simplest type of connectionist network, the linear associator.

The most fundamental operations on binary tree structures, for instance, are those of extracting the left subtree ($\mathbf{ex}_0$), extracting the right subtree ($\mathbf{ex}_1$), and constructing a new tree by embedding two given trees as the left and right subtrees of the new tree (**cons**).

Let $\mathbf{s}$ be the vector realizing $\mathbf{s} = \texttt{[NP VP]}$, with $\mathbf{v}_1$ and $\mathbf{v}_2$ the vectors realizing $\mathbf{s}$'s left and right subtrees $\texttt{NP} = \mathbf{ex}_0\texttt{(s)}$ and $\texttt{VP} = \mathbf{ex}_1\texttt{(s)}$. Then there exist two matrices $\mathbb{W}_{ex0}$ and $\mathbb{W}_{ex1}$ obeying (2.22):

$$\texttt{NP} = \mathbf{ex}_0\texttt{(s)} \qquad \texttt{VP} = \mathbf{ex}_1\texttt{(s)} \tag{2.21}$$

$$\mathbf{v}_1 = \mathbb{W}_{ex0} \cdot \mathbf{s} \qquad \mathbf{v}_2 = \mathbb{W}_{ex1} \cdot \mathbf{s} \tag{2.22}$$

Now let $\mathbf{v}_2$ be the vector realizing $\texttt{VP} = \texttt{[Vt NP]}$, with $\mathbf{v}_3$ and $\mathbf{v}_4$ the vectors realizing $\texttt{VP}$'s left and right subtrees $\texttt{Vt} = \mathbf{ex}_0\texttt{(VP)}$ and $\texttt{NP} = \mathbf{ex}_1\texttt{(VP)}$. If we successively perform a sequence of two extraction operations, first extracting the right child of $\mathbf{s}$, getting $\texttt{VP}$, then extracting the left child of $\texttt{VP}$, we get a structure $\texttt{Vt}$:

$$\texttt{VP} = \mathbf{ex}_1\texttt{(s)} \qquad \texttt{Vt} = \mathbf{ex}_0\texttt{(}\mathbf{ex}_1\texttt{(s))} \tag{2.23}$$

$$\mathbf{v}_2 = \mathbb{W}_{ex1} \cdot \mathbf{s} \qquad \mathbf{v}_3 = \mathbb{W}_{ex0} \cdot (\mathbb{W}_{ex1} \cdot \mathbf{s}) \tag{2.24}$$

By the rules of matrix multiplication, two successive matrix products are equivalent to a single matrix product. So (2.24) can be interpreted as (2.26):

$$\mathbb{W}_{ex0} \cdot (\mathbb{W}_{ex1} \cdot \mathbf{s}) = (\mathbb{W}_{ex0} \cdot \mathbb{W}_{ex1}) \cdot \mathbf{s} \tag{2.25}$$

$$= \mathbb{W}_{ex01} \cdot \mathbf{s} \tag{2.26}$$

where

$$\mathbb{W}_{f\boldsymbol{\rho}} \equiv \mathbb{W}_{f[\boldsymbol{\rho}]_1} \cdot \mathbb{W}_{f[\boldsymbol{\rho}]_2} \cdot \ldots \cdot \mathbb{W}_{f[\boldsymbol{\rho}]_n} \tag{2.27}$$

$$\boldsymbol{\rho} \in \mathbb{B}^n \tag{2.28}$$

This can be recursively repeated to any depth. Thus, the extraction of any subconstituent, no matter how deep in the tree, is achieved by multiplication of a single appropriate matrix. Such a matrix has a special structure – that of a product of instances of the fundamental matrices $\mathbb{W}_{ex0}$ and $\mathbb{W}_{ex1}$.

Like extracting constituents from an existing tree, constructing new binary trees from existing constituents can be performed by linear operations based on two fundamental matrices, $\mathbb{W}_{cons0}$ and $\mathbb{W}_{cons1}$. To return to our example above, the vector realizing the composed tree $\mathbf{s} = \texttt{[NP VP]} = \texttt{[NP [Vt NP]]}$ is:

$$\mathbf{s} = \mathbb{W}_{cons0} \cdot \mathbf{v}_1 + \mathbb{W}_{cons1} \cdot \mathbf{v}_2 \tag{2.29}$$

$$= \mathbb{W}_{cons0} \cdot \mathbf{v}_1 + \mathbb{W}_{cons1} \cdot (\mathbb{W}_{cons0} \cdot \mathbf{v}_3 + \mathbb{W}_{cons1} \cdot \mathbf{v}_4) \tag{2.30}$$

$$= \mathbb{W}_{cons0} \cdot \mathbf{v}_1 + \mathbb{W}_{cons10} \cdot \mathbf{v}_3 + \mathbb{W}_{cons11} \cdot \mathbf{v}_4 \tag{2.31}$$

(2.29) is actually equivalent to (2.6), which defined a recursive connectionist realization. It is repeated as (2.32):

$$\mathbf{s} = \mathbf{v}_1 \otimes \mathbf{r}_0 + \mathbf{v}_2 \otimes \mathbf{r}_1 \tag{2.32}$$

This is because the matrices $\mathbb{W}_{\text{cons0}}$ and $\mathbb{W}_{\text{cons1}}$ are defined so that taking their matrix product with a vector $\mathbf{v}$ achieves the same result as taking the tensor product of $\mathbf{v}$ with $\mathbf{r}_0$ and $\mathbf{r}_1$, respectively.

Definition 2.10 (Kronecker Delta)**.** *The Kronecker δ is defined by*

$$\delta_i^j \equiv \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{2.33}$$

Definition 2.11 (Tensor Product of Matrices)**.** *Let $\mathbb{A}$ and $\mathbb{B}$ be matrices of dimension $m \times n$ and $p \times q$, respectively. Their tensor product $\mathbb{A} \otimes \mathbb{B}$ is the $mp \times nq$ matrix*

$$\mathbb{A} \otimes \mathbb{B} = \begin{bmatrix} a_{11}\mathbb{B} & \cdots & a_{1n}\mathbb{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbb{B} & \cdots & a_{mn}\mathbb{B} \end{bmatrix} \tag{2.34}$$

Claim 2.2 (Matrix Multiplication of Tensor Products)**.** *Let $\mathbb{A}$, $\mathbb{B}$, $\mathbb{C}$, and $\mathbb{D}$ be matrices of dimension $m \times n$, $p \times q$, $n \times r$, and $q \times s$, respectively. Then $(\mathbb{A} \otimes \mathbb{B}) \cdot (\mathbb{C} \otimes \mathbb{D})$ is the $mp \times rs$ matrix*

$$(\mathbb{A} \otimes \mathbb{B}) \cdot (\mathbb{C} \otimes \mathbb{D}) = (\mathbb{A} \cdot \mathbb{C}) \otimes (\mathbb{B} \cdot \mathbb{D}) \tag{2.35}$$

Proof.

$$(\mathbb{A} \otimes \mathbb{B}) \cdot (\mathbb{C} \otimes \mathbb{D}) = \begin{bmatrix} a_{11}\mathbb{B} & \cdots & a_{1n}\mathbb{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbb{B} & \cdots & a_{mn}\mathbb{B} \end{bmatrix} \cdot \begin{bmatrix} c_{11}\mathbb{D} & \cdots & c_{1r}\mathbb{D} \\ \vdots & \ddots & \vdots \\ c_{n1}\mathbb{D} & \cdots & c_{nr}\mathbb{D} \end{bmatrix} \tag{2.36}$$

$$= \begin{bmatrix} \sum_{i=1}^n a_{1i}c_{i1}(\mathbb{B} \cdot \mathbb{D}) & \cdots & \sum_{i=1}^n a_{1i}c_{ir}(\mathbb{B} \cdot \mathbb{D}) \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^n a_{mi}c_{i1}(\mathbb{B} \cdot \mathbb{D}) & \cdots & \sum_{i=1}^n a_{mi}c_{ir}(\mathbb{B} \cdot \mathbb{D}) \end{bmatrix} \tag{2.37}$$

$$= (\mathbb{A} \cdot \mathbb{C}) \otimes (\mathbb{B} \cdot \mathbb{D}) \tag{2.38}$$

□

Corollary 2.1 (Matrix-Vector Multiplication over Tensor Products)**.** *Let $\mathbb{A}$ and $\mathbb{B}$ be matrices of dimension $m \times n$ and $p \times q$, respectively. Let $\mathbf{v}$ and $\mathbf{w}$ be vectors of dimension n and q, respectively. Then $(\mathbb{A} \otimes \mathbb{B}) \cdot (\mathbf{v} \otimes \mathbf{w})$ is the mp-dimensional vector*

$$(\mathbb{A} \otimes \mathbb{B}) \cdot (\mathbf{v} \otimes \mathbf{w}) = (\mathbb{A} \cdot \mathbf{v}) \otimes (\mathbb{B} \cdot \mathbf{w}) \tag{2.39}$$

Proof. Replace $\mathbb{C}$ and $\mathbb{D}$ in (2.36) with $\mathbf{v}$ and $\mathbf{w}$, respectively. (2.39) then follows from an identical argument.

□

Definition 2.12 (Feed-Forward Recursion Matrix)**.** *The **feed-forward recursion matrix** is defined by*

$$\mathbb{I} \equiv 1 + \mathbb{1}_R + \mathbb{1}_R^{\otimes 2} + \mathbb{1}_R^{\otimes 3} + \cdots \tag{2.40}$$

$$= \sum_{k=0}^{\infty} \mathbb{1}_R^{\otimes k} \tag{2.41}$$

$\mathbb{I}$ *is the identity matrix on the total role vector space, including all tree depths.*

Proposition 2.1 (Construction)**.** *The* **cons** *operation for building the realization of a binary tree from that of its two subtrees is defined by*

$$\mathbf{s} = \mathtt{cons}(\mathbf{p}, \mathbf{q}) \Rightarrow \mathbf{s} = \mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1 \tag{2.42}$$

$$= \mathbb{W}_{\text{cons0}} \cdot \mathbf{p} + \mathbb{W}_{\text{cons1}} \cdot \mathbf{q} \tag{2.43}$$

$$\equiv \mathbf{cons}(\mathbf{p}, \mathbf{q}) \tag{2.44}$$

where

$$\mathbb{W}_{\text{cons0}} = \mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{r}_0 \qquad \mathbb{W}_{\text{cons1}} = \mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{r}_1 \tag{2.45}$$

$$\mathbb{I} = \begin{bmatrix} 1 & & & & \cdots \\ & \mathbb{1}_2 & & & \\ & & \mathbb{1}_4 & & \\ & & & \mathbb{1}_8 & \\ \vdots & & & & \ddots \end{bmatrix} \tag{2.46}$$

$$\mathbf{p}, \mathbf{q} \in V_F \tag{2.47}$$

$$\mathbf{r}_0, \mathbf{r}_1 \in V_R \tag{2.48}$$

Proof. Assuming a two-dimensional role vector space V_R, suppose $\mathbf{b}$ is a depth-d binding, $\mathbf{b} \in \mathcal{S}_{(d)}$. For any binding $\mathbf{b} \equiv \mathbf{f}_x \otimes \mathbf{r}_x$, the effect of multiplying $\mathbf{b}$ by $\mathbb{W}_{\text{cons0}}$ must be to produce:

$$\mathbb{W}_{\text{cons0}} \cdot \mathbf{b} \equiv \mathbf{b}' \tag{2.49}$$

$$\equiv \mathbf{f}_x \otimes \mathbf{r}_{x0} \tag{2.50}$$

$$= \mathbf{f}_x \otimes (\mathbf{r}_x \otimes \mathbf{r}_0) \tag{2.51}$$

$$= (\mathbf{f}_x \otimes \mathbf{r}_x) \otimes \mathbf{r}_0 \tag{2.52}$$

$$= \mathbf{b} \otimes \mathbf{r}_0 \tag{2.53}$$

Therefore, $\mathbb{W}_{\text{cons0}} \cdot \mathbf{b} = \mathbf{b}' = \mathbf{b} \otimes \mathbf{r}_0$ is a depth-$d+1$ binding in $\mathcal{S}_{(d+1)}$, with components:

$$[\mathbf{b}']_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} = \Big[[\mathbb{W}_{\text{cons0}} \cdot \mathbf{b}]_{(d+1)}\Big]_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} \tag{2.54}$$

$$= [\mathbf{b} \otimes \mathbf{r}_0]_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} \tag{2.55}$$

$$= [\mathbf{b}]_{\phi\rho_{d+1}\rho_d\cdots\rho_2}[\mathbf{r}_0]_{\rho_1} \tag{2.56}$$

$\mathbb{W}_{\text{cons0}}$ must take $\mathbf{b} \in \mathcal{S}_{(d)}$ into $\mathbf{b}' \in \mathcal{S}_{(d+1)}$ in such a way as to retain all values of components except to multiply each by the ρ_1^{th} component of $\mathbf{r}_0$, adding one additional index ρ_1. Thus, the nonzero elements of the matrix $\mathbb{W}_{\text{cons0}}$ must be:

$$[\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)}]^{\phi\rho_{d+1}\rho_d\cdots\rho_2}_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} = [\mathbf{r}_0]_{\rho_1} \tag{2.57}$$

(2.57) defines $\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)}$, a submatrix of $\mathbb{W}_{\text{cons0}}$ that takes symbols at depth d in $\mathbf{p}$ and puts them at depth $d+1$ in the new tree $\mathbf{s} = \mathbf{cons(p, q)}$. (2.58) shows the block structure of the matrix $\mathbb{W}_{\text{cons0}}$:

$$\mathbb{W}_{\text{cons0}} = \begin{bmatrix} - & - & - & - \\ [\mathbb{W}_{\text{cons0}}]^{(0)}_{(1)} & - & - & - \\ - & [\mathbb{W}_{\text{cons0}}]^{(1)}_{(2)} & - & - \\ - & - & [\mathbb{W}_{\text{cons0}}]^{(2)}_{(3)} & - \\ \vdots & \vdots & - & \ddots \end{bmatrix} \tag{2.58}$$

And so, in the general case:

$$[\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)}]^{\phi'\rho'_{d+1}\rho'_d\cdots\rho'_2}_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} = \delta^{\rho'_{d+1}}_{\rho_{d+1}}\delta^{\rho'_d}_{\rho_d}\cdots\delta^{\rho'_2}_{\rho_2}\delta^{\phi'}_{\phi}[\mathbf{r}_0]_{\rho_1} \tag{2.59}$$

The elements $\{\delta_\phi^{\phi'}\}$ form the identity matrix $\mathbb{1}_F$ in the filler vector space V_F of vectors realizing atoms, while the elements $\{\delta_\rho^{\rho'}\}$ constitue the identity matrix $\mathbb{1}_R$ in the space V_R of role vectors. So (2.59) can be interpreted as:

$$[\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)}]^{\phi'\rho'_{d+1}\rho'_d\cdots\rho'_2}_{\phi\rho_{d+1}\rho_d\cdots\rho_2\rho_1} = [\mathbb{1}_R]^{\rho'_{d+1}}_{\rho_{d+1}}[\mathbb{1}_R]^{\rho'_d}_{\rho_d}\cdots[\mathbb{1}_R]^{\rho'_2}_{\rho_2}[\mathbb{1}_F]^{\phi'}_{\phi}[\mathbf{r}_0]_{\rho_1} \tag{2.60}$$

Using the tensor product of matrices, we can write $\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)}$ more succinctly, by expressing (2.60) as:

$$\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)} = \mathbb{1}_R \otimes \cdots \otimes \mathbb{1}_R \otimes \mathbb{1}_F \otimes \mathbf{r}_0 \tag{2.61}$$

where there are d factors of $\mathbb{1}_R$. We will also write (2.61) as:

$$\mathbb{W}_{\text{cons0}}{}^{(d)}_{(d+1)} = \mathbb{1}_R^{\otimes d} \otimes \mathbb{1}_F \otimes \mathbf{r}_0 \tag{2.62}$$

The full matrix $\mathbb{W}_{\text{cons0}}$ has one such block for each depth d:

$$\begin{aligned}
\mathbb{W}_{\text{cons0}} &= [\mathbb{1}_F \otimes \mathbf{r}_0] + [\mathbb{1}_R \otimes \mathbb{1}_F \otimes \mathbf{r}_0] + [\mathbb{1}_R^{\otimes 2} \otimes \mathbb{1}_F \otimes \mathbf{r}_0] + \cdots && (2.63)\\
&= (1 + \mathbb{1}_R + \mathbb{1}_R^{\otimes 2} + \cdots) \otimes \mathbb{1}_F \otimes \mathbf{r}_0 && (2.64)\\
&= (\sum_{k=0}^{\infty} \mathbb{1}_R^{\otimes k}) \otimes \mathbb{1}_F \otimes \mathbf{r}_0 && (2.65)\\
&= \mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{r}_0 && (2.66)
\end{aligned}$$

We can interpret (2.66) as stating that $\mathbb{W}_{\text{cons0}}$ is the recursive version of the operation of multiplying by $\mathbf{r}_0$. The weight matrix pushes or embeds a tensor realizing a tree into the role of left child of the root of a new tree. The matrix $\mathbb{W}_{\text{cons1}}$ satisfies the analogous equation, with $\mathbf{r}_0$ replaced by $\mathbf{r}_1$. Combining $\mathbb{W}_{\text{cons0}}$ and $\mathbb{W}_{\text{cons0}}$ gives the **cons** operation (2.44).

□

Whereas constructing a new binary tree from two subtrees is defined as the superposition of tensor product bindings, extracting a tree's left or right subtree is equivalent to *unbinding* r_0 or r_1. Assuming the role vectors to be linearly independent, these unbinding operations can be performed accurately via linear operations $\mathbf{ex}_0$ and $\mathbf{ex}_1$: a kind of inner product of $\mathbf{s}$ with an *unbinding vector* $\mathbf{u}_0$ or $\mathbf{u}_1$.

Definition 2.13 (Basis)**.** *A set $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ of n linearly independent vectors in an n-dimensional vector space V is called a **basis** of V.*

Definition 2.14 (Dual Basis)**.** *The basis **dual** to $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ (a set of n linearly independent vectors) is the set $\mathcal{V}^+ = \{\mathbf{v}_1^+, \mathbf{v}_2^+, \ldots, \mathbf{v}_n^+\}$ of (basis) vectors where*

$$\langle \mathbf{v}_i \mid \mathbf{v}_j^+ \rangle = \delta_i^j \tag{2.67}$$

Theorem 2.1 (Unbinding Theorem)**.** *Let* $\mathbf{s}$ *be a tensor product realization induced by a role decomposition with single-valued roles. Suppose the vectors realizing the roles bound in a structure* $\mathtt{s}$ *are all linearly independent. Then each role can be **unbound** with complete accuracy, i.e. for each bound role r_i there is an operation that takes* $\mathbf{s}$ *realizing* $\mathtt{s}$ *into the vector $\mathbf{f}_i$ realizing the filler $\mathtt{f}_i$ bound to r_i.*

Proof. If the role vectors $\{\mathbf{r}_i\}$ are linearly independent, then they form a basis of the subspace of V_R that they span. With respect to this basis, there exists a corresponding dual basis $\{\mathbf{u}_i\} \equiv \{\mathbf{r}_i^+\}$. That is, the inner (dot) product with $\mathbf{u}_i \in V_R$ maps the single role vector $\mathbf{r}_i$ to 1 and all other role vectors to 0. Call $\{\mathbf{u}_i\}$ the unbinding vectors for roles $\{r_i\}$. (The matrix formed by combining all vectors $\{\mathbf{u}_i\}$ is just the inverse of the matrix formed from the vectors $\{\mathbf{r}_i\}$.) Now let $\mathbf{s}$ be the tensor product realization of a structure in which the roles $\{r_i\}$ are respectively bound to the fillers $\{\mathtt{f}_i\}$. Then we can extract $\mathtt{f}_j$ from $\mathbf{s}$, or unbind r_j, by the inner product in V_R of $\mathbf{s}$ with the unbinding vector $\mathbf{u}_j$, defined as follows:

$$\begin{aligned}
\mathbf{s} \cdot \mathbf{u}_j &= \Big(\sum_i \mathbf{f}_i \otimes \mathbf{r}_i\Big) \cdot \mathbf{u}_j && (2.68)\\
&\equiv \sum_i \mathbf{f}_i \langle \mathbf{r}_i \mid \mathbf{u}_j \rangle && (2.69)\\
&= \sum_i \mathbf{f}_i \delta_i^j && (2.70)\\
&= \mathbf{f}_j && (2.71)
\end{aligned}$$

□

Proposition 2.2 (Extraction)**.** *The* $\mathtt{ex}$ *operations for extracting the subtrees of a binary tree are defined by*

$$\begin{aligned}
\mathtt{p} = \mathtt{ex}_0(\mathtt{s}) \Rightarrow \mathbf{p} &= \mathbf{ex}_0(\mathbf{s}) && (2.72)\\
&= \mathbf{ex}_0(\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) && (2.73)\\
&\equiv \mathbb{W}_{\mathrm{ex}0} \cdot (\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) && (2.74)
\end{aligned}$$

$$\mathtt{q} = \mathtt{ex}_1(\mathtt{s}) \Rightarrow \mathbf{q} = \mathbf{ex}_1(\mathbf{s}) \tag{2.75}$$

$$= \mathbf{ex}_1(\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) \tag{2.76}$$

$$\equiv \mathbb{W}_{\text{ex1}} \cdot (\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) \tag{2.77}$$

where

$$\mathbb{W}_{\text{ex0}} = \mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{u}_0^{\mathrm{T}} \qquad \mathbb{W}_{\text{ex1}} = \mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{u}_1^{\mathrm{T}} \tag{2.78}$$

$$\mathbb{I} = \begin{bmatrix} 1 & & & & \cdots \\ & \mathbb{1}_2 & & & \\ & & \mathbb{1}_4 & & \\ & & & \mathbb{1}_8 & \\ \vdots & & & & \ddots \end{bmatrix} \tag{2.79}$$

$$\mathbf{p}, \mathbf{q} \in V_F \tag{2.80}$$

$$\mathbf{r}_0, \mathbf{r}_1 \in V_R \tag{2.81}$$

$$\{\mathbf{u}_i\} \equiv \{\mathbf{r}_i^+\} \tag{2.82}$$

Proof. Assuming a two-dimensional role vector space V_R, the effect of multiplication in $\mathcal{S}^*$ by the matrix $\mathbb{W}_{\text{ex0}}$ must be to map tree depth $d+1$ in $\mathbf{s}$ to depth d in $\mathbf{ex}_0\texttt{(s)}$ for all tree depths:

$$\mathbb{W}_{\text{ex0}} \cdot \mathbf{s} = \mathbb{W}_{\text{ex0}} \cdot (\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) \tag{2.83}$$

$$= (\mathbb{I} \otimes \mathbb{1}_F \otimes \mathbf{u}_0^{\mathrm{T}}) \cdot (\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) \tag{2.84}$$

$$= ([\mathbb{I} \otimes \mathbb{1}_F] \otimes \mathbf{u}_0^{\mathrm{T}}) \cdot (\mathbf{p} \otimes \mathbf{r}_0 + \mathbf{q} \otimes \mathbf{r}_1) \tag{2.85}$$

$$= ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{p}) \otimes (\mathbf{u}_0^{\mathrm{T}} \cdot \mathbf{r}_0) + ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{q}) \otimes (\mathbf{u}_0^{\mathrm{T}} \cdot \mathbf{r}_1) \tag{2.86}$$

$$= ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{p}) \langle \mathbf{u}_0 \mid \mathbf{r}_0 \rangle + ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{q}) \langle \mathbf{u}_0 \mid \mathbf{r}_1 \rangle \tag{2.87}$$

$$= ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{p})\delta_0^0 + ([\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{q})\delta_0^1 \tag{2.88}$$

$$= [\mathbb{I} \otimes \mathbb{1}_F] \cdot \mathbf{p} \tag{2.89}$$

$$= \mathbf{p} \tag{2.90}$$

The matrix $\mathbb{W}_{\text{ex1}}$ satisfies the analogous argument, with $\mathbf{u}_0$ replaced by $\mathbf{u}_1$.

□

A crucial property of the primitive matrices realizing the **cons** and **ex** operations is that that their form – $\mathbb{I} \otimes \mathbb{M}$ – is preserved under matrix multiplication. For instance, an **ex** operation followed by a **cons** operation gives (2.94):

$$\mathbb{W}_{\text{cons}i} \cdot \mathbb{W}_{\text{ex}j} = (\mathbb{I} \otimes [\mathbb{1}_F \otimes \mathbf{r}_i]) \cdot (\mathbb{I} \otimes [\mathbb{1}_F \otimes \mathbf{u}_j^{\mathrm{T}}]) \tag{2.91}$$

$$= (\mathbb{I} \cdot \mathbb{I}) \otimes ([\mathbb{1}_F \otimes \mathbf{r}_i] \cdot [\mathbb{1}_F \otimes \mathbf{u}_j^{\mathrm{T}}]) \tag{2.92}$$

$$= (\mathbb{I} \cdot \mathbb{I}) \otimes [(\mathbb{1}_F \cdot \mathbb{1}_F) \otimes (\mathbf{r}_i \cdot \mathbf{u}_j^{\mathrm{T}})] \tag{2.93}$$

$$= \mathbb{I} \otimes [\mathbb{1}_F \otimes (\mathbf{r}_i \cdot \mathbf{u}_j^{\mathrm{T}})] \tag{2.94}$$

In fact, this result is general.

Theorem 2.2 (Feed-Forward Recursion Theorem)**.** *Suppose* $\mathbb{W} = \mathbb{I} \otimes \underline{\mathbb{W}}$ *and* $\mathbb{X} = \mathbb{I} \otimes \underline{\mathbb{X}}$*, where* $\underline{\mathbb{W}}$ *and* $\underline{\mathbb{X}}$ *are matrices of dimension* $m \times n$ *and* $p \times q$*, respectively. Then*

$$\mathbb{W} \cdot \mathbb{X} = \mathbb{I} \otimes \underline{\mathbb{Y}} \tag{2.95}$$

where

$$\underline{\mathbb{Y}} \equiv \begin{cases} \underline{\mathbb{W}} \cdot [\mathbb{1}^{\otimes \Delta} \otimes \underline{\mathbb{X}}] & \Delta \equiv n - p > 0 \\ [\mathbb{1}^{\otimes \Delta'} \otimes \underline{\mathbb{W}}] \cdot \underline{\mathbb{X}} & \Delta' \equiv p - n > 0 \\ \underline{\mathbb{W}} \cdot \underline{\mathbb{X}} & \Delta = \Delta' = 0 \end{cases} \tag{2.96}$$

Proof. (2.95) and (2.96) follow from (2.35) and (2.41).

□

$\mathbb{I}$ takes the finite number of elements in $\underline{\mathbb{W}}$ that govern rearrangements near the root of a tree and 'copies' them unboundedly to fill up $\mathbb{W}$, creating a matrix that performs the same rearrangement at unbounded depths in a tree. The idealization to unboundedly deep trees realized in unbounded (or infinite) networks or $\mathbb{W}$ matrices is immediate. The infinite behavior of recursive function evaluation is generated through the finite specification (or knowledge) $\underline{\mathbb{W}}$.

The significance of the feed-forward recursion theorem is that we have a means of constructing simple networks which compute arbitrarily complex symbolic (recursive) functions. We can prove that they do indeed correctly compute such functions, and therefore we can explain with complete precision and certainty how they do so. Yet such an explanation is not possible simply by means of symbolic algorithms. Explanations must use the connectionist algorithm of a linear associator (2.19), with a weight matrix possessing special global structure.

As an example, consider a structure-sensitive function **f** that maps an input tree **s** to its corresponding output tree **t**:

$$\begin{aligned}\mathbf{f(s)} = \mathbf{cons}(&\mathbf{ex}_1(\mathbf{ex}_0(\mathbf{ex}_1(\mathbf{s}))),\\ &\mathbf{cons}(\mathbf{ex}_1(\mathbf{ex}_1(\mathbf{ex}_1(\mathbf{s}))),\mathbf{ex}_0(\mathbf{s})))\end{aligned} \tag{2.97}$$

f takes as input the tree structure of an English sentence in the passive voice and produces as output a tree structure encoding a predicate calculus form of the semantic interpretation of the input sentence:

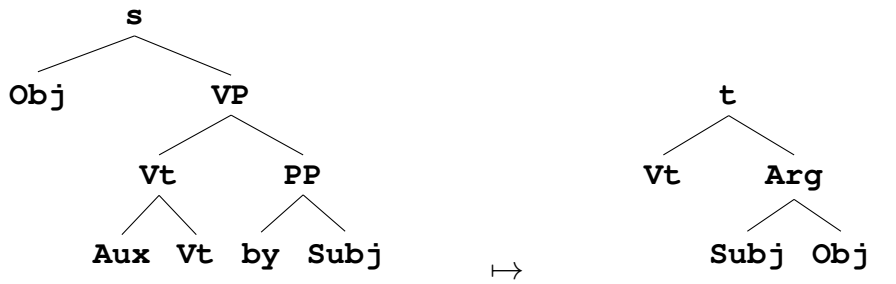


Figure 2.4: **f :: s -> t**

The connectionist realization of the symbol-manipulating function **f** is a sequence of interleaved extraction and construction operations combined into a single composite matrix $\mathbb{W}$:

$$\begin{aligned}\mathbb{W} = &\ \mathbb{W}_{\text{cons0}} \cdot (\mathbb{W}_{\text{ex1}} \cdot \mathbb{W}_{\text{ex0}} \cdot \mathbb{W}_{\text{ex1}})\\ &+ \mathbb{W}_{\text{cons1}} \cdot (\mathbb{W}_{\text{cons0}} \cdot (\mathbb{W}_{\text{ex1}} \cdot \mathbb{W}_{\text{ex1}} \cdot \mathbb{W}_{\text{ex1}}) + \mathbb{W}_{\text{cons1}} \cdot (\mathbb{W}_{\text{ex0}}))\end{aligned} \tag{2.98}$$

The fact that **f** is correctly realized by (2.98) is explained by the fact that $\mathbb{W}$ possesses certain global structure: $\mathbb{W}$ is a certain product and sum of fundamental matrices $\mathbb{W}_{\text{ex0}}$, $\mathbb{W}_{\text{ex1}}$, $\mathbb{W}_{\text{cons0}}$, and $\mathbb{W}_{\text{cons1}}$. The correspondence between the internal structure of the expressions for **f** and $\mathbb{W}$ is evident.

The example above provides a simple illustration of the ICS computational perspective. At a lower level of analysis, we have a connectionist network in which mental representations are distributed over many processing units, and outputs are computed from inputs by massively parallel spreading activation. At a higher level of analysis, the global structure inherent in the activation vectors, and the weight matrices that process them, entails that the network, $\mathbb{W}$, provably computes a particular recursive function, $\mathtt{f}$. This function can be described with a symbolic expression, as in (2.97), but the network does not compute the function in the step-by-step fashion standardly used to interpret such expressions. Rather, all the symbol manipulation is done in one step, in a massively parallel and distributed fashion. The computational resources required by the processing correspond to the number of units and weights needed, the precision with which individual units must operate, and the number of steps of activation spreading required to compute the output (i.e. 1) – not the number of symbols used, or the number of symbolic operations performed in interpreting the symbolic expression.

(2.98) is representative of one important class of connectionist networks: feed-forward networks, where there are no closed loops of activation flow. The other class consists of the networks that do have such loops: *recurrent networks*. Of particular interest is a subclass of these networks which perform optimization: *harmonic networks*. Harmonic networks can realize essential kinds of symbolic functions, including those determined by grammars. Like the weight matrices of feed-forward networks, the weight matrices of recurrent networks have a special structure.

Definition 2.15 (Dualizer Matrix)**.** *The **dualizer** matrix is defined by*

$$\mathbb{D} = \begin{bmatrix} \mathbf{d}_0 & \mathbf{d}_1 \end{bmatrix}^{\mathrm{T}} \tag{2.99}$$

where

$$\mathbf{d}_0 \equiv [\mathbf{u}_0]_0 \mathbf{u}_0 + [\mathbf{u}_1]_0 \mathbf{u}_1 \qquad \mathbf{d}_1 \equiv [\mathbf{u}_0]_1 \mathbf{u}_0 + [\mathbf{u}_1]_1 \mathbf{u}_1 \tag{2.100}$$

Claim 2.3 (Symmetry of Dualizer Matrix)**.** $\mathbb{D}$ *is symmetric, i.e.*

$$\mathbb{D}^{\mathrm{T}} = \mathbb{D} \tag{2.101}$$

Proof.

$$\mathbb{D}^{\mathrm{T}} \equiv \begin{bmatrix} \mathbf{d}_0 & \mathbf{d}_1 \end{bmatrix} \tag{2.102}$$

$$= \begin{bmatrix} [\mathbf{d}_0]_0 & [\mathbf{d}_1]_0 \\ [\mathbf{d}_0]_1 & [\mathbf{d}_1]_1 \end{bmatrix} \tag{2.103}$$

and

$$\mathbb{D} \equiv \begin{bmatrix} \mathbf{d}_0 & \mathbf{d}_1 \end{bmatrix}^{\mathrm{T}} \tag{2.104}$$

$$= \begin{bmatrix} [\mathbf{d}_0]_0 & [\mathbf{d}_0]_1 \\ [\mathbf{d}_1]_0 & [\mathbf{d}_1]_1 \end{bmatrix} \tag{2.105}$$

Hence, $\mathbb{D}^{\mathrm{T}} = \mathbb{D}$ if and only if $[\mathbf{d}_0]_1 = [\mathbf{d}_1]_0$. But,

$$[\mathbf{d}_0]_1 \equiv [[\mathbf{u}_0]_0 \mathbf{u}_0 + [\mathbf{u}_1]_0 \mathbf{u}_1]_1 \tag{2.106}$$

$$= [\mathbf{u}_0]_0 [\mathbf{u}_0]_1 + [\mathbf{u}_1]_0 [\mathbf{u}_1]_1 \tag{2.107}$$

and

$$[\mathbf{d}_1]_0 \equiv [[\mathbf{u}_0]_1 \mathbf{u}_0 + [\mathbf{u}_1]_1 \mathbf{u}_1]_0 \tag{2.108}$$

$$= [\mathbf{u}_0]_1 [\mathbf{u}_0]_0 + [\mathbf{u}_1]_1 [\mathbf{u}_1]_0 \tag{2.109}$$

$$= [\mathbf{d}_0]_1 \tag{2.110}$$

□

Claim 2.4 (Dualizer)**.** *Suppose $|x| = k$, where $k > 0$. Then*

$$\mathbb{D}^{\otimes k} \cdot \mathbf{r}_x = \mathbf{u}_x \tag{2.111}$$

($\mathbb{D}$ is an '$\mathbf{r}$ to $\mathbf{r}^+ \equiv \mathbf{u}$ converter'.)

Proof. (Base case) Suppose $k = 1$. Since row i of $\mathbb{D}$ is $\mathbf{d}_i^{\mathrm{T}}$, the i^{th} component of $\mathbb{D} \cdot \mathbf{r}_0$ is

$$[\mathbb{D} \cdot \mathbf{r}_0]_i = \mathbf{d}_i^{\mathrm{T}} \cdot \mathbf{r}_0 \tag{2.112}$$

$$= ([\mathbf{u}_0]_i \mathbf{u}_0 + [\mathbf{u}_1]_i \mathbf{u}_1)^{\mathrm{T}} \cdot \mathbf{r}_0 \tag{2.113}$$

$$= [\mathbf{u}_0]_i (\mathbf{u}_0^{\mathrm{T}} \cdot \mathbf{r}_0) + [\mathbf{u}_1]_i (\mathbf{u}_1^{\mathrm{T}} \cdot \mathbf{r}_0) \tag{2.114}$$

$$= [\mathbf{u}_0]_i \langle \mathbf{u}_0 \mid \mathbf{r}_0 \rangle + [\mathbf{u}_1]_i \langle \mathbf{u}_1 \mid \mathbf{r}_0 \rangle \tag{2.115}$$

$$= [\mathbf{u}_0]_i \delta_0^0 + [\mathbf{u}_1]_i \delta_1^0 \tag{2.116}$$

$$= [\mathbf{u}_0]_i \tag{2.117}$$

Hence, $\mathbb{D} \cdot \mathbf{r}_0 = \mathbf{u}_0$. An identical argument shows $\mathbb{D} \cdot \mathbf{r}_1 = \mathbf{u}_1$. (Induction) Assume (2.111) holds for some value of k. Suppose $\mathbf{r}_x = \mathbf{r}_{iy}$, where $|x| = k + 1$, $|y| = k$, and $|i| = 1$. Then

$$\mathbb{D}^{\otimes (k+1)} \cdot \mathbf{r}_x = (\mathbb{D} \otimes \mathbb{D}^{\otimes k}) \cdot \mathbf{r}_{iy} \tag{2.118}$$

$$\equiv (\mathbb{D} \otimes \mathbb{D}^{\otimes k}) \cdot (\mathbf{r}_i \otimes \mathbf{r}_y) \tag{2.119}$$
$$= (\mathbb{D} \cdot \mathbf{r}_i) \otimes (\mathbb{D}^{\otimes k} \cdot \mathbf{r}_y) \tag{2.120}$$
$$= \mathbf{u}_i \otimes \mathbf{u}_y \tag{2.121}$$
$$\equiv \mathbf{u}_{iy} \tag{2.122}$$
$$\equiv \mathbf{u}_x \tag{2.123}$$

□

Definition 2.16 (Recurrent Recursion Matrix)**.** *The **recurrent recursion matrix** is defined by*

$$\mathbb{R} \equiv 1 + \mathbb{D} + \mathbb{D}^{\otimes 2} + \mathbb{D}^{\otimes 3} + \cdots \tag{2.124}$$
$$= \sum_{k=0}^{\infty} \mathbb{D}^{\otimes k} \tag{2.125}$$

Definition 2.17 (Recursive Weight Matrix)**.** *The weight matrix $\mathbb{W}$ of a symmetric recurrent network is **recursive** if it obeys the recursion equation*

$$\mathbb{W} = \underline{\mathbb{W}} + \mathbb{W} \otimes \mathbb{D} \tag{2.126}$$

*where $\underline{\mathbb{W}}$ is a finite **root matrix** such that for all $\mathbf{C}$, $\mathbf{t}$,*

$$\mathbf{C} / \mathbf{r}_x^{\mathrm{T}} \cdot \underline{\mathbb{W}} \cdot \mathbf{t} = 0 \tag{2.127}$$

unless $x = \varepsilon$ (i.e. root position).

Theorem 2.3 (Recursive Weight Matrix Theorem)**.** *A **recursive weight matrix** $\mathbb{W}$ of a symmetric recurrent network has the form*

$$\mathbb{W} = \underline{\mathbb{W}} \otimes \mathbb{R} \tag{2.128}$$

where $\underline{\mathbb{W}}$ is a root matrix and $\mathbb{R}$ is the recurrent recursion matrix, which solves the recursion equation

$$\mathbb{R} = 1 + \mathbb{R} \otimes \mathbb{D} \tag{2.129}$$

Proof. Substituting (2.126) into itself recursively yields

$$\mathbb{W} = [\underline{\mathbb{W}}] + \mathbb{W} \otimes \mathbb{D} \tag{2.130}$$
$$= [\underline{\mathbb{W}}] + (\underline{\mathbb{W}} + \mathbb{W} \otimes \mathbb{D}) \otimes \mathbb{D} \tag{2.131}$$
$$= [\underline{\mathbb{W}}] + [\underline{\mathbb{W}} \otimes \mathbb{D}] + \mathbb{W} \otimes \mathbb{D} \otimes \mathbb{D} \tag{2.132}$$

$$= [\underline{\mathbb{W}}] + [\underline{\mathbb{W}} \otimes \mathbb{D}] + (\underline{\mathbb{W}} + \mathbb{W} \otimes \mathbb{D}) \otimes \mathbb{D} \otimes \mathbb{D} \tag{2.133}$$

$$= [\underline{\mathbb{W}}] + [\underline{\mathbb{W}} \otimes \mathbb{D}] + [\underline{\mathbb{W}} \otimes \mathbb{D} \otimes \mathbb{D}] + \mathbb{W} \otimes \mathbb{D} \otimes \mathbb{D} \otimes \mathbb{D} \tag{2.134}$$

$$= \cdots \tag{2.135}$$

$$= [\underline{\mathbb{W}}] + [\underline{\mathbb{W}} \otimes \mathbb{D}] + [\underline{\mathbb{W}} \otimes \mathbb{D} \otimes \mathbb{D}] + \cdots \tag{2.136}$$

$$= \underline{\mathbb{W}} \otimes (1 + \mathbb{D} + \mathbb{D}^{\otimes 2} + \cdots) \tag{2.137}$$

$$\equiv \underline{\mathbb{W}} \otimes \mathbb{R} \tag{2.138}$$

An analogous argument verifies that $\mathbb{R}$ itself solves $\mathbb{R} = 1 + \mathbb{R} \otimes \mathbb{D}$. □

Central aspects of many higher cognitive domains (including language) are realized via recursive processing. Feed-forward networks and recurrent networks provide a mechanism to compute a large class of cognitive functions with recursive structure. In either case, $\underline{\mathbb{W}}$ is a finite matrix of weights that specifies a particular function. $\mathbb{I}$ and $\mathbb{R}$ are the respective recursion matrices. These are simply-defined unbounded matrices that are fixed – i.e. the same for all cognitive functions – as shown in (2.41) and (2.125).

2.3 Harmony

In a number of cognitive domains, information processing in the mind/brain constructs an output for which the pair **`(input,output)`** is *optimal*. This kind of processing maximizes a connectionist well-formedness measure called *harmony*. The harmony function encapsulates knowledge as a set of conflicting soft constraints of varying strengths. Harmony maximization ensures that the output achieves the optimal degree of simultaneous satisfaction of these constraints.

The harmony of an activation vector in a connectionist network is a numerical measure of the degree to which the vector respects the constraints encoded in the connection matrix, i.e. the degree to which the completion of an input vector is well-formed, according to the network's connections. The result of spreading activation, therefore, is the generation of a pattern of activity that best satisfies a set of parallel soft constraints. The constraints are 'soft' in the sense that each may be overruled by other constraints, and hence violated in the final pattern. The cost of constraint violation is the lowering of harmony, by an amount depending on the strength of the violated constraint and the degree of violation.

Definition 2.18 (Harmony of an Activation Vector)**.** *The **harmony** of an activation vector* $\mathbf{a}$ *is defined by*

$$H(\mathbf{a}) = \sum_{\beta\alpha} H_{\beta\alpha} \tag{2.139}$$

$$= \sum_{\beta\alpha} a_\beta W_{\beta\alpha} a_\alpha \tag{2.140}$$

$$= \mathbf{a}^T \cdot \mathbb{W} \cdot \mathbf{a} \tag{2.141}$$

where $\sum_{\beta\alpha}$ *means 'sum over all pairs of units* β*,* α*' and* $\mathbb{W}$ *is a recursive weight matrix encoding a set of network constraints, i.e.* $\mathbb{W} = \underline{\mathbb{W}} \otimes \mathbb{R}$*.*

The importance of harmony is that it allows connectionist principles to inform higher levels of analysis. Using representations subserving higher cognition which possess tensor product structure, we can derive a symbolic interpretation of (2.141).

Proposition 2.3 (Harmony of a Symbolic Representation)**.** *Suppose* $\mathbf{a}$ *is a tensor product vector realizing a symbolic structure* $\mathbf{s}$ *with constituents* $\{\mathbf{c}_i\}$*. Then*

$$H(\mathbf{s}) \equiv H(\mathbf{a}) \tag{2.142}$$

$$= \sum_{i \leq j} H(\mathbf{c}_i, \mathbf{c}_j) \tag{2.143}$$

where $H(\mathbf{c}_i, \mathbf{c}_j)$ *– the harmony resulting from the co-occurrence of* $\mathbf{c}_i$ *and* $\mathbf{c}_j$ *in the same structure – is a constant for all* $\mathbf{s}$*.*

Proof. Let $\{\mathbf{c}_i\}$ be the vectors realizing the constituents $\{\mathbf{c}_i\}$ (filler/role bindings) of a symbolic structure $\mathbf{s}$ realized by $\mathbf{a}$. Then

$$H(\mathbf{a}) = \mathbf{a}^T \cdot \mathbb{W} \cdot \mathbf{a} \tag{2.144}$$

$$= (\sum_i \mathbf{c}_i)^T \cdot \mathbb{W} \cdot (\sum_i \mathbf{c}_i) \tag{2.145}$$

$$= \sum_k \mathbf{c}_k^T \cdot \mathbb{W} \cdot \mathbf{c}_k + \sum_{i \neq j} \mathbf{c}_i^T \cdot \mathbb{W} \cdot \mathbf{c}_j \tag{2.146}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + \sum_{i \neq j} \mathbf{c}_i^T \cdot \mathbb{W} \cdot \mathbf{c}_j \tag{2.147}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + [\sum_{i<j} \mathbf{c}_i^T \cdot \mathbb{W} \cdot \mathbf{c}_j + \sum_{i>j} \mathbf{c}_i^T \cdot \mathbb{W} \cdot \mathbf{c}_j] \tag{2.148}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + [\sum_{i<j} \mathbf{c}_i^T \cdot \mathbb{W} \cdot \mathbf{c}_j + \sum_{i>j} \mathbf{c}_j^T \cdot \mathbb{W}^T \cdot \mathbf{c}_i] \tag{2.149}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + [\sum_{i<j} \mathbf{c}_i^{\mathrm{T}} \cdot \mathbb{W} \cdot \mathbf{c}_j + \sum_{j>i} \mathbf{c}_i^{\mathrm{T}} \cdot \mathbb{W}^{\mathrm{T}} \cdot \mathbf{c}_j] \tag{2.150}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + [\sum_{i<j} \mathbf{c}_i^{\mathrm{T}} \cdot (\mathbb{W} + \mathbb{W}^{\mathrm{T}}) \cdot \mathbf{c}_j] \tag{2.151}$$

$$= \sum_k H(\mathbf{c}_k, \mathbf{c}_k) + \sum_{i<j} H(\mathbf{c}_i, \mathbf{c}_j) \tag{2.152}$$

$$= \sum_{i \leq j} H(\mathbf{c}_i, \mathbf{c}_j) \tag{2.153}$$

where

$$H(\mathbf{c}_k, \mathbf{c}_k) \equiv \mathbf{c}_k^{\mathrm{T}} \cdot \mathbb{W} \cdot \mathbf{c}_k \qquad H(\mathbf{c}_i, \mathbf{c}_j) \equiv \mathbf{c}_i^{\mathrm{T}} \cdot (\mathbb{W} + \mathbb{W}^{\mathrm{T}}) \cdot \mathbf{c}_j \tag{2.154}$$

□

(2.143) reveals that the harmony a symbolic structure $\mathbf{s}$ is simply the summation of the individual harmonies arising from the co-occurrence of constituents in $\mathbf{s}$. In other words, if $\mathbf{s}$ contains the constituents $\mathbf{c}_i$ and $\mathbf{c}_j$, then add the numerical quantity $H(\mathbf{c}_i, \mathbf{c}_j)$ to $H(\mathbf{s})$. Furthermore, (2.152) shows that the individual harmonies in a symbolic structure can be divided into two groups: $H(\mathbf{c}_k, \mathbf{c}_k)$ and $H(\mathbf{c}_i, \mathbf{c}_j)$, where $i \neq j$. $H(\mathbf{c}_k, \mathbf{c}_k)$ can be interpreted as the *self-harmony* of $\mathbf{c}_k$: a measure of the degree to which the internal structure of the activation pattern $\mathbf{c}_k$ realizing the symbolic constituent $\mathbf{c}_k$ conforms to the soft constraints embodied in the network's connections. $H(\mathbf{c}_i, \mathbf{c}_j)$, where $i \neq j$, can be interpreted as the *interaction harmony* of the pair $(\mathbf{c}_i, \mathbf{c}_j)$: the additional harmony (positive or negative) arising from the co-presence of $\mathbf{c}_i$ and $\mathbf{c}_j$ beyond the sum of their individual harmonies in isolation. This result is a simple kind of (automatic) *compositionality*: the harmony of a structure as a whole is the sum of the harmonies contributed by all of its constituents. What's remarkable is the sheer simplicity of the computation. In the case where the quantities $H(\mathbf{c}_i, \mathbf{c}_j)$ are known, no reference to the connectionist description's activation vectors and weight matrices is required! The entire computation can be performed on the basis of the symbolic constituents alone, since the numbers $H(\mathbf{c}_i, \mathbf{c}_j)$ encapsulate the details of the lower-level weights and activations into a form directly usable at the higher-level.

2.4 Harmonic Grammar

Among the cognitive domains falling under harmony maximization are central aspects of knowledge of language – grammar. In this setting, the specification of

the harmony function is called a *harmonic grammar*. So (2.143) can be interpreted as a collection of *soft rules*. These rules define a harmonic grammar. To then determine the harmony of a structure $\mathbf{s}$, we simply find all the rules that apply to $\mathbf{s}$ and add up all the corresponding harmony contributions.

Definition 2.19 (Harmonic Grammar)**.** *Let R_{ij} be a soft rule defined by*

$$\begin{aligned} R_{ij} \equiv \textit{If } \mathbf{s} \textit{ simultaneously contains the constituents } \mathbf{c}_i \textit{ and } \mathbf{c}_j, \\ \textit{then add the numerical quantity } H(\mathbf{c}_i, \mathbf{c}_j) \textit{ to } H(\mathbf{s}). \end{aligned} \tag{2.155}$$

Then the collection of rules $\{R_{ij}\}$ define a ***harmonic grammar****. Equivalently, soft rules can be recast as soft constraints. Let C_{ij} be a soft constraint defined by*

$$C_{ij} \equiv \begin{cases} \mathbf{s} \textit{ does not simultaneously contain} \\ \textit{the constituents } \mathbf{c}_i \textit{ and } \mathbf{c}_j & H(\mathbf{c}_i, \mathbf{c}_j) < 0 \\ \textit{(with strength } w_{ij}) \\ \\ \mathbf{s} \textit{ simultaneously contains} \\ \textit{the constituents } \mathbf{c}_i \textit{ and } \mathbf{c}_j & H(\mathbf{c}_i, \mathbf{c}_j) \geq 0 \\ \textit{(with strength } w_{ij}) \end{cases} \tag{2.156}$$

where

$$w_{ij} \equiv |H(\mathbf{c}_i, \mathbf{c}_j)| \tag{2.157}$$

Then the collection of constraints $\{C_{ij}\}$ define a ***harmonic grammar****. In this formulation, $H(\mathbf{s})$ is computed by adding the strengths of all the positive constraints that $\mathbf{s}$ satisfies and subtracting the strengths of all the negative constraints that it violates.*

As an example, consider the context-free rewrite-grammar rule expressing a sentence (**S**) as a noun phrase (**NP**) followed by a verb phrase (**VP**):

$$\mathbf{S} \rightarrow \mathbf{NP} \;\; \mathbf{VP} \tag{2.158}$$

This rule asserts the grammaticality of the local tree:

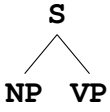


Figure 2.5: Graphical representation of soft rules

Two corresponding soft rules, for instance, are:

1. $R_{\mathtt{NP/S}} \equiv$ If $\mathtt{s}$ contains a constituent labeled $\mathtt{S}$ and its left subconstituent is labeled $\mathtt{NP}$, then add $+2$ to H.

2. $R_{\mathtt{S\backslash VP}} \equiv$ If $\mathtt{s}$ contains a constituent labeled $\mathtt{S}$ and its right subconstituent is labeled $\mathtt{VP}$, then add $+4$ to H. (Verb phrases are usually considered *first-class* citizens of sentences.)

That these rules instantiate the general soft rule schema for R_{ij} is evident. In $R_{\mathtt{NP/S}}$, for example, the constituent $\mathtt{c}_i$ is an $\mathtt{S}$ at some node in the parse tree, while the second constituent $\mathtt{c}_j$ is an $\mathtt{NP}$ at a node that is the left-child node of the $\mathtt{S}$. The harmony contribution from this pair, $H(\mathtt{c}_i, \mathtt{c}_j)$, is $+2$. That this quantity is positive means that this pair of constituents is well formed according to the grammar encoded in the network's connections $\mathbb{W}$.

Given the ability to assess the harmony of any symbolic structure – which, of course, are realized by tensor product representations – it is now (in principle) straightforward to determine the functioning of a grammar. An harmonic grammar assigns to an input vector $\mathbf{i}$ the output symbolic structure ('parse') $\mathtt{s}$ with maximal harmony, among those which are completions of $\mathbf{i}$. This computation is achieved via activation flow from the input units to other units, whereby each unit repeatedly updates its activation value in response to input from other units. Typically, this eventually settles and the overall activation vector $\mathbf{a}$ of the entire network, no longer changes. The input vector $\mathbf{i}$ (and also the output vector $\mathbf{o}$) is part of the total activation vector $\mathbf{a}$, since the input units (and the output units) are part of the total population of units in the network. For context-free grammars, the parse is a tree whose terminal symbols give the input string, and whose structure shows how the input symbols are grouped into constituent phrases. For formal grammars, we can arbitrarily impose a threshold of acceptable harmony (e.g. 0) for the parse in order for an input string to be judged sufficiently well-formed by the grammar to be admitted into the language.

If I have seen further than others, it is by standing on the shoulders of giants.

— Isaac Newton

3 Compositional Distributional Model of Semantics

The *Integrated Connectionist/Symbolic Architecture* (ICS) presented in [52] is a compelling proposal to resolve the problem of competing symbolic and connectionist models of mind. While the former are compositional but only qualitative, the latter are non-compositional but quantiative. ICS manages to combine the two approaches and therefore (in principle) get the best out of both paradigms. However, as we have seen, ICS does admit some weaknesses. Not only do symbolic structures exist in an unbounded representational space $\mathcal{S}^*$, but also symbolic structures of the same type (e.g. a sentence) can manifest different representations, i.e. live in conflicting mental spaces, depending on the grammar of the respective instances.

Interestingly, a similar problem has been encountered within the context of natural language processing (NLP) concerning the unification of compositional and distributional theories of semantics [8]. [7, 12] introduce a mathematical formalism aimed at solving this problem. The approach involves adapting a category-theoretic model, originally used to describe information flow in quantum mechanics (QM), to the task of composing semantic vectors. Category theory provides the sufficient tools to relate the structures used to represent grammatical types, i.e. *pregroup grammars*, and the structures used to represent distributional meaning, i.e. *vector spaces*, which are both concrete instances of a *compact closed category*. As such, the framework enables the computation of the meaning of a well-typed sentence from the meaning of its constituents. Importantly, meanings of whole sentences live in a single space, independent of the grammatical structure of the sentence. Hence, the *inner product* can be used to compare meanings of arbitrary sentences, as it is for comparing the meaning of words in distributional models.

Here, we introduce the compositional distributional (DISCO) model of semantics presented in [7, 12] along with its diagrammatic calculus and recent extension regarding the modelling of pronouns [47]. Our proposal is that the interaction of QM and ICS is an encouraging new direction for the computational modelling of the mind, of which the DISCO framework forms the basis. This work provides the second abutment of the bridge – linking cognitive science and quantum mechanics – which we ultimately aim to build.

3.1 Competing Models of Semantics

First, we give a brief overview of two prominent domains of computational linguistics: compositional semantics and distributional semantics. Whereas compositional models are based on logical formalisms, distributional models represent meaning as vectors within an empirically learned semantic space. The difference between these two approaches reduces to a simple question concerning the foundational structure of semantics: Is meaning *functional* or *contextual*?

3.1.1 Compositional Semantics

Compositional semantics provides methods for mapping natural language to logical formulae. The key idea behind formal models is that the meaning of a sentence is a function of its words and their syntax. Lambda expressions are used to model meanings of words as functions, and function application serves as the composition operation which relates the meanings of individual words and returns the meaning of a larger phrase. In layman's terms, the mapping from natural language to logic consists of pairing syntactic analysis rules with semantic interpretation rules (and applying β-reduction where necessary).

There is much appeal to compositional semantics. For one, a formal approach to linguistics captures the desirable feature of compositionality. What's more, the model-theoretic basis of formal semantics provides the required framework to characterize the notion of a true sentence and the notion of entailment [39]. The denotation of a sentence characterizes in which worlds a sentence is true, and entailment spans all possible worlds.

That being said, compositional semantics leaves much to be desired. While logical analysis does produce $\{true, false\}$-valuations, no notion of semantics is communicated beyond truth conditions. In this sense, compositional semantics reduces the richness of natural language to a single bit ($\in \mathbb{B}$), overlooking more

complex and realistic notions of semantic representation. Furthermore, as with any logic, a description of symbols and valuation functions must be given. This leaves open the question of how we might *learn* the meaning of natural language using such a model, rather than merely use it.

Syntactic Analysis	Semantic Interpretation
S $\rightarrow$ **NP VP**	**VP(NP)**
NP $\rightarrow$ **Adj NP**	**Adj(NP)**
NP $\rightarrow$ *comedians, jokes, book, etc.*	*comedians, etc.*
Adj $\rightarrow$ *funny, loud, bright, etc.*	$\lambda x.funny(x)$, *etc.*
VP $\rightarrow$ **Vt NP**	**Vt(NP)**
Vt $\rightarrow$ *tell, bring, make, etc.*	$\lambda y.\lambda x.tell(x,y)$, *etc.*

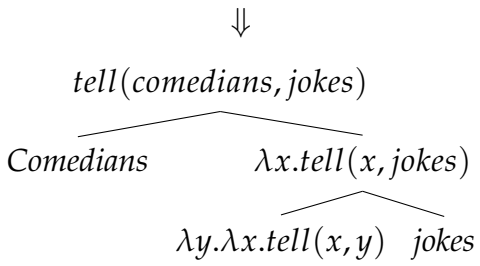


Figure 3.1: From natural language to logic

3.1.2 Distributional Semantics

In contrast to compositional semantics, distributional semantics represents meanings of words as vectors in a high-dimensional semantic space. Distributional representations are computed via analysis of word co-occurrence statistics which are learned from corpora. More concretely, the meaning of a word is defined by the words with which it occurs and the degree with which it occurs with said words within a certain context. In this way, word meaning is truly distributional: the semantic content of a word corresponds to the pattern in which language spreads it among a set of context words. Hence, the meaning of a word is just a sample of its true distribution within language.

In simple models, (orthogonal) basis vectors of the semantic space make up a set $\{\mathbf{c}_i\}_i$ of context words, where $\mathbf{c} \in \mathbb{R}^n$ and $n = |\{\mathbf{c}_i\}_i|$. The representation of a given word is then a vector $\mathbf{w}$ in the semantic space $\mathbb{R}^n$ where the i^{th} component $[\mathbf{w}]_i$ is the normalized co-occurrence (i.e. frequency) count of the words $\mathbf{w}$ and $\mathbf{c}_i$ describe within a certain context window evaluated over a collection of texts. The

semantic vector of a word can therefore be characterized as the weighted superposition of basis vectors, i.e. $\mathbf{w} = \sum_i c_i \mathbf{c}_i$ where $c_i \in \mathbb{R}$.

There are several advantages to distributional semantics. For one, meaning can be learned. What's more, unlike pure logic, vector representations are able to capture complex notions of semantic similarity. For example, a basic measure of similarity between two (normalized) word representations $\mathbf{w}_1$ and $\mathbf{w}_2$ is easily modelled as an inner product, or cosine measure, in the semantic space, i.e. $\langle \mathbf{w}_1 \mid \mathbf{w}_2 \rangle \in [0, 1]$.

word	*stand-up*	*funny*	*actor*
comedian	1000	1000	500
joke	2500	1250	0
improv	100	0	350

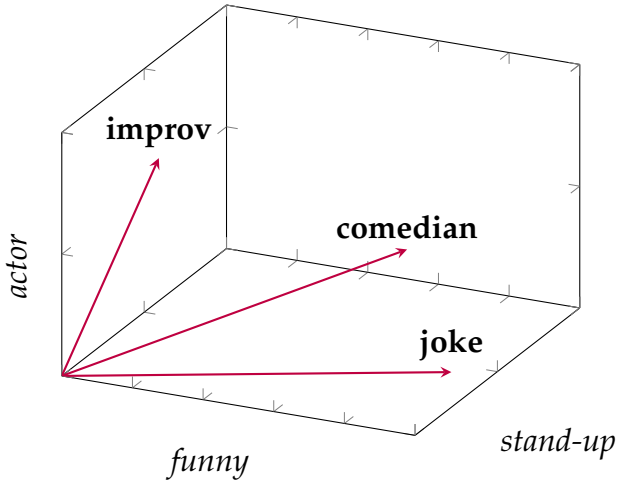
$$\Downarrow$$

$$\mathbf{w} = c_{stand\text{-}up}\mathbf{stand\text{-}up} + c_{funny}\mathbf{funny} + c_{actor}\mathbf{actor}$$

$$\Downarrow$$

$$\{\mathbf{comedian} = \begin{bmatrix}\frac{2}{3} & \frac{2}{3} & \frac{1}{3}\end{bmatrix}^{\mathrm{T}}, \mathbf{joke} = \begin{bmatrix}\frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0\end{bmatrix}^{\mathrm{T}}, \mathbf{improv} = \begin{bmatrix}\frac{2}{\sqrt{53}} & 0 & \frac{7}{\sqrt{53}}\end{bmatrix}^{\mathrm{T}}\}$$

$$\Downarrow$$



$$\Downarrow$$

$$\langle \mathbf{comedian} \mid \mathbf{improv} \rangle = \frac{11}{3\sqrt{53}}$$

Figure 3.2: From natural language to vector spaces

Be that as it may, the distributional approach to linguistics does have weaknesses. Distributional semantics operates almost entirely at the word level. This is because there is no practical method to collect meaningful co-occurrence statistics

at the sentence level. Rather, to go from word vectors to sentence vectors we must provide a composition operation. However, no obvious procedure exists. Another limitation of the distributional model is the lack of any notion of entailment. Interestingly, the weaknesses of distributional semantics are precisely the strengths of compositional semantics and vice versa. It is in this sense that the compositional and distributional approaches to semantics have traditionally been viewed to be orthogonal.

3.2 Pregroup Grammar

Pregroup grammars are a class of type-logical grammar [32, 33]. A *pregroup grammar* consists of atomic grammatical types which can combine to freely generate compound types. A series of application rules permit *type reductions* (i.e. composition) and *type introductions*. These rules form the basis of syntactic analysis.

Definition 3.1 (Partially Ordered Monoid)**.** *A* ***partially ordered monoid*** $(P, \leq, \cdot, 1)$ *is a partially ordered set, equipped with a monoid multiplication* $- \cdot -$ *with unit 1 where for* $p, q, r \in P$

$$p \leq q \quad \Rightarrow \quad r \cdot p \leq r \cdot q \quad \text{and} \quad p \cdot r \leq q \cdot r \tag{3.1}$$

Definition 3.2 (Pregroup)**.** *A* ***pregroup*** $(P, \leq, \cdot, 1, (-)^l, (-)^r)$ *is a partially ordered monoid whose each element* $p \in P$ *has a left adjoint* p^l *and a right adjoint* p^r *obeying the inequalities*

$$p^l \cdot p \leq 1 \leq p \cdot p^l \quad \text{and} \quad p \cdot p^r \leq 1 \leq p^r \cdot p \tag{3.2}$$

Let us unravel this definition. P is simply a set of objects $\{p, q, r, \ldots\}$. $\leq$ denotes a partial ordering on P, i.e. an ordering relation between the elements of P. $- \cdot -$ is an associative, non-commutative monoid multiplication operator which can be thought of as a function $- \cdot - : P \times P \to P$. For instance, if $p, q \in P$, then we have $p \cdot q \in P$. This means that P is *closed* under this operation. $1 \in P$ is the unit, satisfying $p \cdot 1 = 1 = 1 \cdot p$ for $p \in P$, and is self-adjoint, i.e. $1^r = 1 = 1^l$. Finally, $(-)^l$ and $(-)^r$ are the adjoint generators, which can be thought of as functions $(-)^l : P \to P$ and $(-)^r : P \to P$ such that for $p \in P$, $p^l, p^r \in P$. Adjoints are further described by a number of properties:

1. Adjoints are unique.

2. Adjoints are order reversing: $p \leq q \Rightarrow q^r \leq p^r$ and $q^l \leq p^l$.

3. Multiplication is self-adjoint: $(p \cdot q)^r = q^r \cdot p^r$ and $(p \cdot q)^l = q^l \cdot p^l$.

4. Opposite adjoints annihilate each other: $(p^l)^r = p = (p^r)^l$.

5. Same adjoints iterate: $p^{ll} \cdot p^l \leq 1 \leq p^{rr} \cdot p^r, p^{lll} \cdot p^{ll} \leq 1 \leq p^{rrr} \cdot p^{rr}, \ldots$.

Pregroups are useful because they allow us to formalize grammar. (Henceforth, we use an arrow $\rightarrow$ for $\leq$ and drop $- \cdot -$ between juxtaposed types.) First, we fix a set of basic grammatical roles, e.g. $\{n$ (noun), s (sentence), $\ldots\}$, and a partial ordering between them. Next, we freely generate a pregroup of these types, e.g. $\{\ldots, nn^l$ (adjective), $n^r s n^l$ (transitive verb), $\ldots\}$, via application of adjoints and juxtaposition to form compound types from atomic types. A type (atomic or compound) is then assigned to each word of a lexicon depending on its syntactic role. We define a string of words to be grammatical if there exists a reduction from the type of the expression to the basic type s, i.e. sentence.

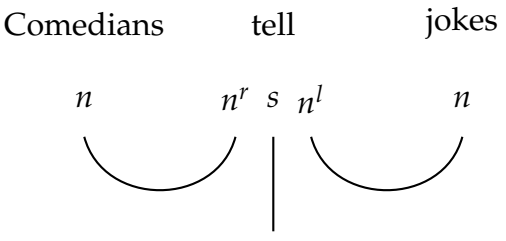
As an example, consider the sentence *Comedians tell jokes*. *Comedians* and *jokes* are nouns of type n, and *tell* is a transitive verb of type $n^r s n^l$.[1] Therefore, the type of *Comedians tell jokes* is:

$$\begin{array}{ccc} \text{Comedians} & \text{tell} & \text{jokes} \\ n & (n^r s n^l) & \text{n} \end{array} \tag{3.3}$$

and the expression is grammatical because of the following reduction:

$$n(n^r s n^l) n \rightarrow (n n^r) s (n^l n) \rightarrow 1 s 1 \rightarrow s \tag{3.4}$$

We can depict reductions more conveniently using a diagrammatic notation. Reductions are represented by underlinks between contracting types (and introductions are represented by overlinks between expanding types). So (3.3) and (3.4) together can be interpreted as:



(3.5)

[1] When dealing with compound types, it is helpful to consider an adjoint p^r as expressing, "Expects type p on the left", and an adjoint p^l as expressing, "Expects type p on the right" (and vice versa).

(3.5) illustrates rather neatly how noun arguments compose with a transitive verb to produce a sentence. What's more, we can see how word order contributes to semantics: *Comedians* on the left is the *subject* argument and *jokes* on the right is the *object* argument.

3.3 Basic Category Theory

The mathematical model of language we present is category-theoretic. Category theory is a branch of pure mathematics which formalizes mathematical structure and its concepts in terms of a collection of *objects* and *morphisms*. Its simplicity and compact conceptual language make category theory both general and specific. Categorical axioms allow the deduction of new specific properties of existing theories, and shared categorical representations enable communication between properties of unconnected theories. The power of category theory is precisely its capacity to pass information within and across mathematical structures. The DISCO model of semantics harnesses this potential in order to unify the seemingly orthogonal compositional and distributional models of semantics.

3.3.1 Monoidal Categories

Definition 3.3 (Monoidal Category)**.** *A (strict)* ***monoidal category*** $\mathbf{C}$ *is defined by*

1. *A family* $|\mathbf{C}|$ *of object such that*
   (a) *For each ordered pair of objects* (A, B)*, there is a corresponding set* $\mathbf{C}(A, B)$ *of morphisms. We denote* $f \in \mathbf{C}(A, B)$ *by* $f : A \to B$*.*
   (b) *For each ordered triple of objects* (A, B, C)*, each* $f : A \to B$*, and* $g : B \to C$*, there is a sequential composite* $g \circ f : A \to C$*. We moreover required that*
   $$(h \circ g) \circ f = h \circ (g \circ f) \tag{3.6}$$
   (c) *For each object* A*, there is an identity morphism* $1_A : A \to A$*. For* $f : A \to B$ *we moreover require that*
   $$f \circ 1_A = f \qquad 1_B \circ f = f \tag{3.7}$$
2. *For each ordered pair of objects* (A, B)*, there is a composite object* $A \otimes B$*. We moreover require that*
   $$(A \otimes B) \otimes C = A \otimes (B \otimes C) \tag{3.8}$$

3. *There is a unit object* I *which satisfies*

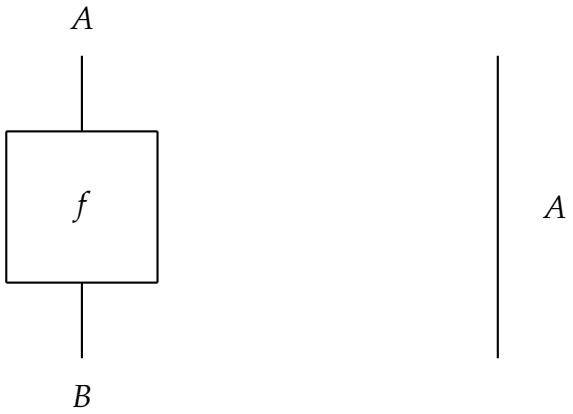
$$\mathrm{I} \otimes A = A = A \otimes \mathrm{I} \tag{3.9}$$

4. *For each ordered pair of morphisms* $(f : A \to C, g : B \to D)$*, there is a parallel composite* $f \otimes g : A \otimes B \to C \otimes D$*. We moreover require bifunctoriality, i.e.*

$$(g_1 \otimes g_2) \circ (f_1 \otimes f_2) = (g_1 \circ f_1) \otimes (g_2 \circ f_2) \tag{3.10}$$

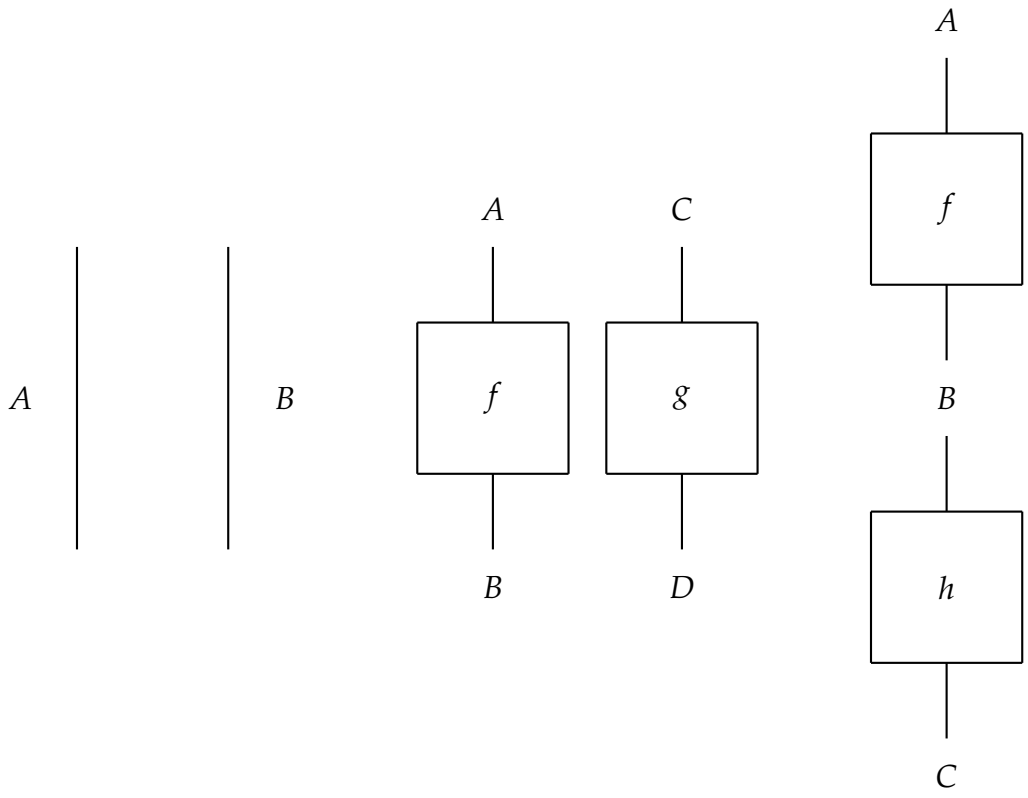
Monoidal categories admit an intuitive operational interpretation. We can think of objects as *types of systems* and morphisms as *processes* which take systems of one type (i.e. input) to systems of another type (i.e. output). Composition of morphisms is then just *sequential application* of processes, and compound types, e.g. $A \otimes B$, represent *joint systems*. We can think of I as the trivial system, i.e. 'nothing', and so the morphisms $\beta : \mathrm{I} \to A$ and $\theta : A \to \mathrm{I}$ correspond to *states* (or *elements*) of A and *co-states* of A, respectively.

An interesting property of monoidal categories is that they are equipped with a beautiful, purely diagrammatic calculus in the sense that a provable equational statement between morphisms in a monoidal category must be derivable in the graphical language [51]. In fact, this is true not only for monoidal categories, but also for those with additional structure, including compact closed categories (which we will encounter shortly). In this graphical calculus, we depict morphisms as boxes, with incoming and outgoing wires labelled with corresponding types. Sequential composition is conveyed via connecting matching outputs and inputs, and parallel composition is illustrated by juxtaposing boxes side by side. Identity is seen as a naked wire, and the unit is an empty diagram, i.e. 'nothing'. (Following convention, we depict information flow from top to bottom.) For example, the morphism $f : A \to B$ and an object A with the identity $1_A : A \to A$ are depicted as:



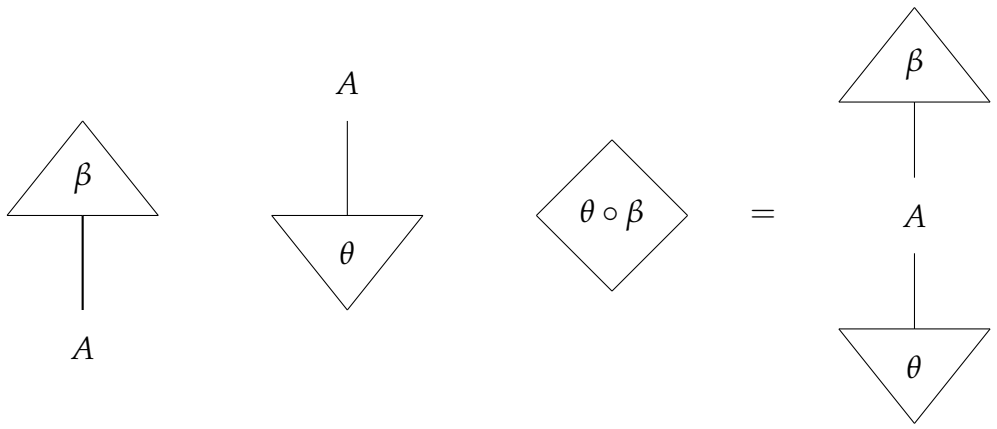
(3.11)

The object $A \otimes B$ and the morphisms $f \otimes g$ and $h \circ f$, where $f : A \to B$, $g : C \to D$, and $h : B \to C$, are depicted as:



(3.12)

The morphisms $\beta : \mathrm{I} \to A$, $\theta : A \to \mathrm{I}$, and $\theta \circ \beta : \mathrm{I} \to \mathrm{I}$, are depicted as:



(3.13)

3.3.2 Compact Closed Categories

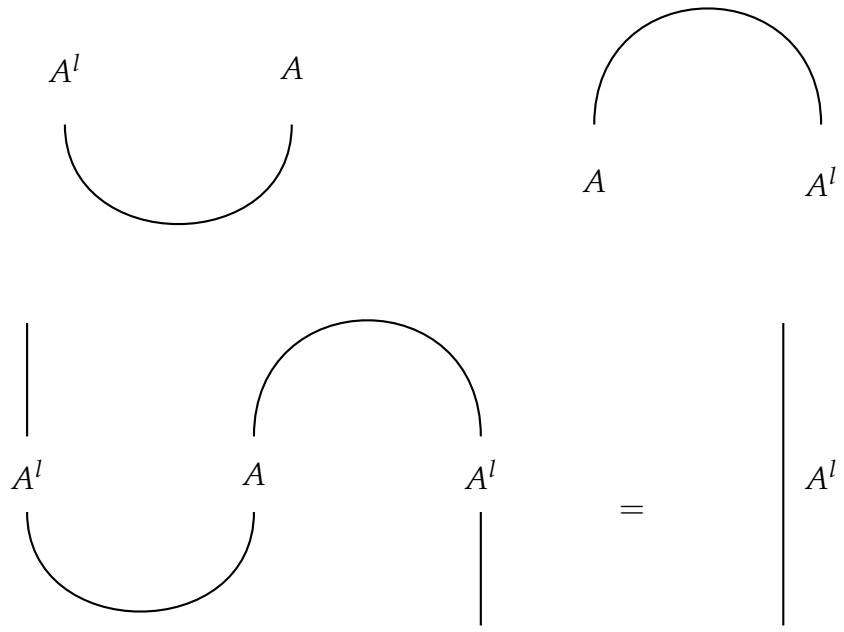
Definition 3.4 (Compact Closed Category)**.** *A monoidal category is* ***compact closed*** *if for each object A there are also objects A^r, A^l, and morphisms*

$$\eta^l : \mathrm{I} \to A \otimes A^l \quad \epsilon^l : A^l \otimes A \to \mathrm{I} \quad \eta^r : \mathrm{I} \to A^r \otimes A \quad \epsilon^r : A \otimes A^r \to \mathrm{I} \tag{3.14}$$

which satisfy (the 'yanking' equations)

$$(1_A \otimes \epsilon^l) \circ (\eta^l \otimes 1_A) = 1_A \qquad (\epsilon^r \otimes 1_A) \circ (1_A \otimes \eta^r) = 1_A$$
$$(\epsilon^l \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l) = 1_{A^l} \qquad (1_{A^r} \otimes \epsilon^r) \circ (\eta^r \otimes 1_{A^r}) = 1_{A^r} \tag{3.15}$$

Diagrammatically, ϵ maps are depicted by 'cups' (i.e. underlinks) and η maps by 'caps' (i.e. overlinks). Their composition boils down to 'yanking wires'. For instance, $\epsilon^l : A^l \otimes A \to \mathrm{I}$, $\eta^l : \mathrm{I} \to A \otimes A^l$, and $(\epsilon^l \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l) = 1_{A^l}$ are depicted as:



(3.16)

3.4 A Pregroup as a Compact Closed Category

By the compositional hypothesis, the particular model of meaning we want to use is type-logical. Let P be a pregroup. P is an example of a *posetal category*, i.e. a category which is also a partially ordered set. For any two objects in P there is either one or no morphism between them. For example, the morphism $A \to B$ is written as $A \leq B$. For 'objects' $p, q, r \in P$, $[p \leq q]$ is the singleton $\{p \leq q\}$ whenever $p \leq q$ and the empty set otherwise. Furthermore, if $p \leq q$ and $q \leq r$, $p \leq r$ is naturally defined as the composite of the 'morphisms' $p \leq q$ and $q \leq r$. In addition, P is equipped with associative monoid multiplication which behaves as tensor on objects. As such, for $p \leq r$ and $q \leq s$, we have $p \cdot q \leq r \cdot s$ by monotonicity of monoid multiplication. Again, $p \cdot q \leq r \cdot s$ is naturally defined as the tensor of

'morphisms' $[p \leq r]$ and $[q \leq s]$. Moreover, we define 1 as the monoidal unit and note that bifunctoriality is trivially satisfied since there can only be one morphism between any two objects. In sum, the underlying category is given by the partial order, and its monoidal structure is induced by the anatomy of the pregroup.

Proposition 3.1 (Compact Closure in P)**.** *P is a **compact closed category**.*

Proof. Recalling that each object $p \in P$ has unique left and right adjoints, P is a compact closed category for

$$\begin{aligned} \eta^l &= [1 \leq p \cdot p^l] & \eta^r &= [1 \leq p^r \cdot p] \\ \epsilon^l &= [p^l \cdot p \leq 1] & \epsilon^r &= [p \cdot p^r \leq 1] \end{aligned} \tag{3.17}$$

The axioms of compact closure are trivially satisfied. Examining the first 'yanking' equation, we have

$$\begin{aligned} (1 \otimes \epsilon^l) \circ (\eta^l \otimes 1) :: p &= 1 \cdot p && (3.18)\\ &\leq (p \cdot p^l) \cdot p && (3.19)\\ &= p \cdot (p^l \cdot p) && (3.20)\\ &\leq p \cdot 1 && (3.21)\\ &= p && (3.22) \end{aligned}$$

Similar calculations verify that the other 'yanking' equations also hold. Note that, diagrammatically, the underlinks (and overlinks) representing type reductions (and introductions) in pregroup grammars are exactly the 'cups' (and 'caps') of the compact closed structure.

□

3.5 Vector Spaces, Linear Maps, and Tensor Product as a Compact Closed Category

By the distributional hypothesis, the particular model of meaning we want to use is vector spatial. Let **FVect** be the category of vector spaces over the base field $\mathbb{R}$ with linear maps (between vectors spaces) as morphisms, the vector space tensor product as the monoidal tensor, and the base field $\mathbb{R}$ as the monoidal unit. Furthermore, let us assume that each vector space V comes with an inner product operation $\langle \ | \ \rangle : V \times V \to \mathbb{R}$ (a fixed basis canonically induces an inner product). In **FVect**, the tensor is commutative, i.e. $V \otimes W \cong W \otimes V$, since any tensor

is isomorphic to its permutations. Moreover, adjoints are degenerate in **FVect**, i.e. $V^l = V^r$, so we denote either by V^*, which is the identity map, i.e. $V^* = V$. This is because the adjoint of a vector space is its co-vector space – the elements of which are the conjugate transpose of the elements of the original vector space – but the conjugate transpose of a real-valued vector is just its transpose.

Proposition 3.2 (Compact Closure in **FVect**). **FVect** *is a* ***compact closed category****.*

Proof. Given a vector space V with basis $\{\mathbf{e}_i\}_i$, we can verify that compact closure arises via setting $V^l = V^r = V$

$$\eta^l = \eta^r : \mathbb{R} \to V \otimes V :: 1 \mapsto \sum_i \mathbf{e}_i \otimes \mathbf{e}_i \tag{3.23}$$

and

$$\epsilon^l = \epsilon^r : V \otimes V \to \mathbb{R} :: \sum_{ij} c_{ij} \mathbf{v}_i \otimes \mathbf{w}_j \mapsto \sum_{ij} c_{ij} \langle \mathbf{v}_i \mid \mathbf{w}_j \rangle \tag{3.24}$$

Now consider the third 'yanking' equation. Letting $\mathbf{v} \in V$, we have

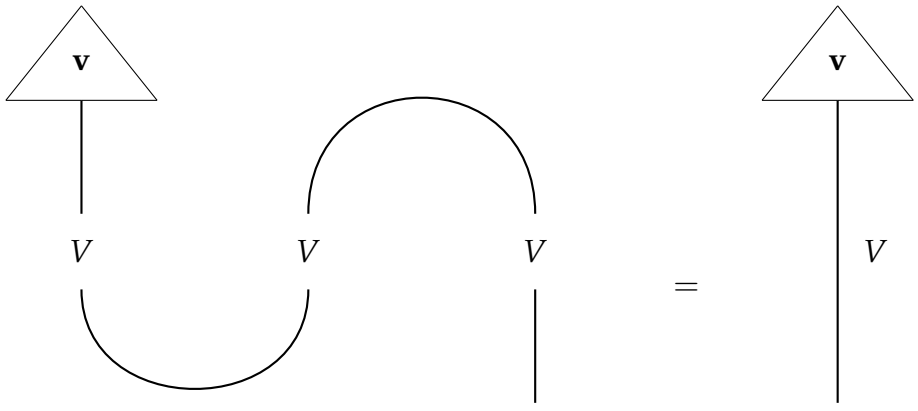
$$(\epsilon^l \otimes 1_V) \circ (1_V \otimes \eta^l) :: \mathbf{v} \mapsto \mathbf{v} \otimes (\sum_i \mathbf{e}_i \otimes \mathbf{e}_i) \tag{3.25}$$

$$= \sum_i (\mathbf{v} \otimes \mathbf{e}_i) \otimes \mathbf{e}_i \tag{3.26}$$

$$\mapsto \sum_i \langle \mathbf{v} \mid \mathbf{e}_i \rangle \mathbf{e}_i \tag{3.27}$$

$$= \mathbf{v} \tag{3.28}$$

Diagrammatically, we can interpret (3.25) $\to$ (3.28) as:



(3.29)

Similar calculations verify that the other 'yanking' equations also hold.

□

3.6 From Syntax to Semantics

Up until this point, we have described two aspects of language in terms of mathematical structures which realize compact closure: *grammar* and *meaning*. Pregroups can be used to characterize the grammatical architecture underlying a language, and vector spaces can be used to assign meanings to words of a language. It is useful to think of these two compact closed categories as structures we can *project* out of language (P is the free pregroup generated from the atomic types of a natural language):

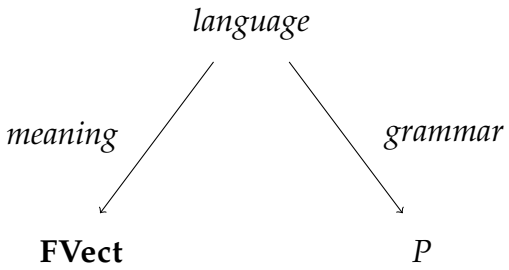


Figure 3.3: Projecting grammar and meaning out of language

The aim of the DisCo model of semantics is to unify both of these aspects of language. This requires either the compositional structure of pregroups to lift to the level of assigning meaning to sentences and their constituents, or dually, a mechanism to compute the meaning of a sentence from the structure of assigning meaning to words. The symmetry of **FVect**, i.e. $V^l = V^r = V$, renders the singular compact closed structure too primitive for this task. What's more, the existence of canonical isomorphisms $V \otimes W \to W \otimes V$ implies that meaning is preserved under different word ordering – in general, (at least in English language) we model meaning to be dependent on word order (e.g. we expect different semantic interpretations of *Comedians tell jokes* and *Jokes tell comedians*). A method sophisticated enough to combine grammar and language must refine types to retain the full grammatical content obtained from the pregroup analysis. [12] proposes a simple solution to the problem: rather than objects in **FVect**, we consider objects in the product category **FVect** $\times P$. Specifically, **FVect** $\times P$ is the category which has pairs (V, p), with V a vector space and $p \in P$, as objects, and pairs $(f, \leq) : (V, p) \to (W, q)$, with f a linear map and $\leq$ a pregroup ordering relation, as morphisms, which we can also write as $(f : V \to W, p \leq q)$. There exists a morphism $(f, \leq) : (V, p) \to (W, q)$ only if there exists a morphism $f : V \to W$ in **FVect** and an ordering relation $p \leq q$ in P.

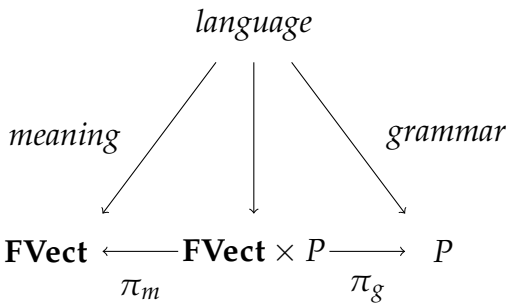


Figure 3.4: **FVect** $\times$ P

Since both **FVect** and P are compact closed, it is straightforward to show that **FVect** $\times$ P is as well. The compact closed structure of **FVect** and P lifts componentwise to the product category **FVect** $\times$ P.

Proposition 3.3 (Compact Closure in **FVect** $\times$ P)**.** **FVect** $\times$ P *is a* ***compact closed category****.*

Proof. Given a vector space V with basis $\{\mathbf{e}_i\}_i$, and recalling that each object $p \in P$ has unique left and right adjoints, **FVect** $\times$ P is a compact closed category for

$$\begin{aligned} &(\eta^l, \leq) : (\mathbb{R}, 1) \to (V \otimes V, p \cdot p^l) \qquad (\eta^r, \leq) : (\mathbb{R}, 1) \to (V \otimes V, p^r \cdot p) \\ &(\epsilon^l, \leq) : (V \otimes V, p^l \cdot p) \to (\mathbb{R}, 1) \qquad (\epsilon^r, \leq) : (V \otimes V, p \cdot p^r) \to (\mathbb{R}, 1) \end{aligned} \tag{3.30}$$

Now consider the second 'yanking' equation. Letting $\mathbf{v} \in V$, we have

$$((\epsilon^r, \leq) \otimes (1_V, 1)) \circ ((1_V, 1) \otimes (\eta^r, \leq)) :: (\mathbf{v}, p) \mapsto (\mathbf{v} \otimes (\sum_i \mathbf{e}_i \otimes \mathbf{e}_i), p \cdot (p^r \cdot p)) \tag{3.31}$$

$$= (\sum_i (\mathbf{v} \otimes \mathbf{e}_i) \otimes \mathbf{e}_i, (p \cdot p^r) \cdot p) \tag{3.32}$$

$$\mapsto (\sum_i \langle \mathbf{v} \mid \mathbf{e}_i \rangle \mathbf{e}_i, 1 \cdot p) \tag{3.33}$$

$$= (\mathbf{v}, p) \tag{3.34}$$

Analogous computations verify that the other 'yanking' equations also hold.

□

FVect $\times$ P provides a mathematical structure rich enough to integrate grammar and meaning in natural language such that we can compute the meaning of a sentence from the meanings of words. In fact, this procedure is just a morphism in **FVect** $\times$ P.

Definition 3.5 (Meaning Space)**.** *We refer to an object (W, p) of* **FVect** $\times P$ *as a* ***meaning space****. This consists of a vector space W in which the meaning of a word lives* $\mathbf{w} \in W$ *and the grammatical type p of the word.*

Definition 3.6 (From Syntax to Semantics)**.** *We define the vector $\overrightarrow{w_1 \cdots w_n}$ of the meaning of a string of words $w_1 \cdots w_n$ to be*

$$\overrightarrow{w_1 \cdots w_n} \equiv f(\mathbf{w}_1 \otimes \cdots \otimes \mathbf{w}_n) \tag{3.35}$$

where for (W_i, p_i) meaning space of the word w_i, the linear map f is built by substituting each p_i in $[p_1 \cdots p_n \leq s]$ with W_i.

Thus, for $\Delta = [p_1 \cdots p_n \to s]$ a morphism in P and $f = \Delta[p_i \backslash W_i]$ a linear map in **FVect***, the following is a morphism in* **FVect** $\times P$*:*

$$(W_1 \otimes \cdots \otimes W_n, p_1 \cdots p_n) \xrightarrow{(f, \leq)} (S, s) \tag{3.36}$$

We call f the 'from-meanings-of-words-to-meaning-of-a-sentence' map.

The key idea behind the linear map f – taking us from meanings of words to meaning of a sentence – is that the pregroup reductions guide the order in which the compact closure maps are applied to the vector spaces. In other words, *syntax guides semantics*.

Example 3.1 (Simple Positive Transitive Sentence)**.** Consider the sentence

Comedians tell jokes. (3.37)

Comedians and *jokes* are both nouns of type n, and so $\mathbf{Comedians}, \mathbf{jokes} \in N$. *tell* is a (positive) transitive verb of type $n^r s n^l$, and so $\mathbf{tell} \in N \otimes S \otimes N$.[2] Therefore, the linear map f encodes the following structural morphism in **FVect** $\times P$:

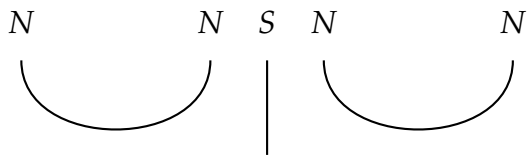
$$(N \otimes (N \otimes S \otimes N) \otimes N, n(n^r s n^l) n) \xrightarrow{(f, \leq)} (S, s) \tag{3.38}$$

and arises from the syntactic reduction map (3.39):

$$f = \epsilon_N \otimes 1_S \otimes \epsilon_N : N \otimes (N \otimes S \otimes N) \otimes N \to S \tag{3.39}$$

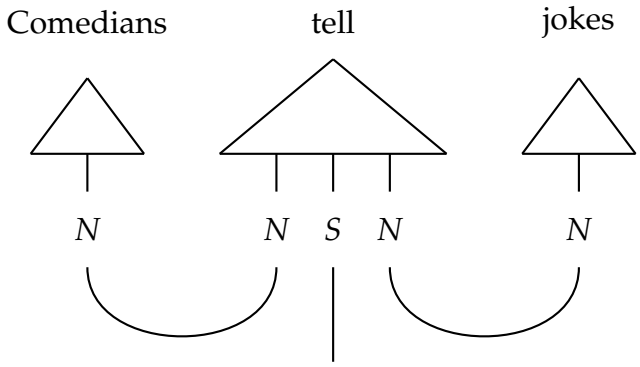
In graphical notation, the linear map of meaning f is depicted as:

[2] We can think of the meaning vector of a transitive verb as a function that inputs a subject from N, an object from N, and outputs a sentence S.



(3.40)

The matrix of f has $dim(N)^2 \times dim(S) \times dim(N)^2$ columns and $dim(S)$ rows, and its entries are either 0 or 1. When applied to the vectors of the meanings of the words, i.e. $f(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) \in S$, we obtain:



(3.41)

Letting $\mathbf{Comedians} = \sum_i c_i \mathbf{n}_i$, $\mathbf{tell} = \sum_{jkl} c_{jkl} \mathbf{n}_j \otimes \mathbf{s}_k \otimes \mathbf{n}_l$, and $\mathbf{jokes} = \sum_m c_m \mathbf{n}_m$, where $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ are orthonormal bases of N and S respectively, we have

$$\overrightarrow{\textit{Comedians tell jokes}} \equiv f(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) \tag{3.42}$$

$$= \epsilon_N \otimes 1_S \otimes \epsilon_N(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) \tag{3.43}$$

$$= \epsilon_N \otimes 1_S \otimes \epsilon_N((\sum_i c_i \mathbf{n}_i) \otimes (\sum_{jkl} c_{jkl} \mathbf{n}_j \otimes \mathbf{s}_k \otimes \mathbf{n}_l) \otimes (\sum_m c_m \mathbf{n}_m)) \tag{3.44}$$

$$= \epsilon_N \otimes 1_S \otimes \epsilon_N(\sum_{ijklm} c_i c_{jkl} c_m \mathbf{n}_i \otimes \mathbf{n}_j \otimes \mathbf{s}_k \otimes \mathbf{n}_l \otimes \mathbf{n}_m) \tag{3.45}$$

$$= \sum_{ijklm} c_i c_{jkl} c_m \langle \mathbf{n}_i \mid \mathbf{n}_j \rangle \mathbf{s}_k \langle \mathbf{n}_l \mid \mathbf{n}_m \rangle \tag{3.46}$$

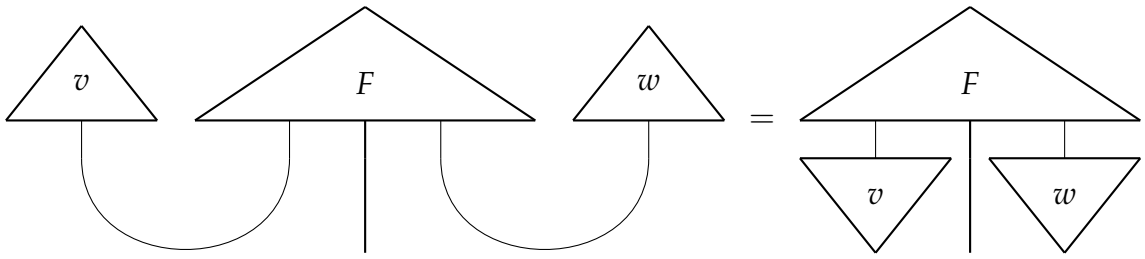
$$= \sum_{ijklm} c_i c_{jkl} c_m \delta_i^j \mathbf{s}_k \delta_l^m \tag{3.47}$$

$$= \sum_k \mathbf{s}_k \sum_{il} c_i c_{ikl} c_l \tag{3.48}$$

$$\in S \tag{3.49}$$

where (3.46) $\rightarrow$ (3.47) makes use of the fact that $\{\mathbf{n}_i\}_i$ forms an orthonormal basis of N.

Finally, we note that the diagrammatic calculus tells us that:



(3.50)

We call this the 'swing' rewrite rule. This simplifies the expression we need to compute. With respect to our example (3.37), the right-hand side of (3.50) corresponds to (3.46).

Example 3.2 (Complex Positive Transitive Sentence)**.** Consider the sentence

$$\textit{Comedians tell funny jokes.} \tag{3.51}$$

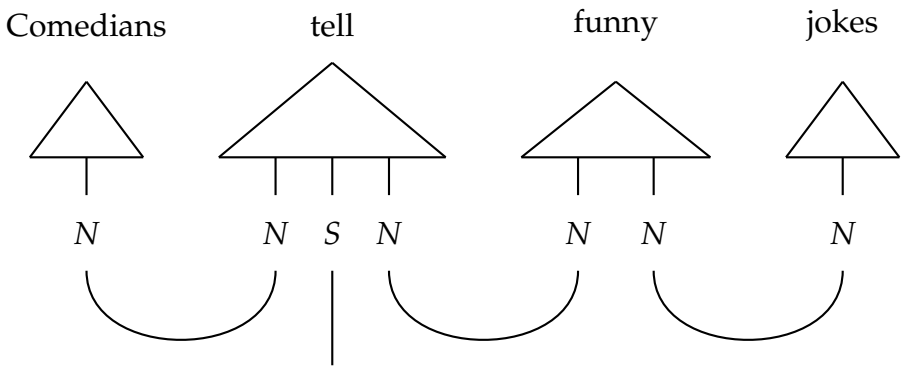
Following on from the previous example, and recognizing that *funny* is an adjective of type nn^l with $\mathbf{funny} \in N \otimes N$, the linear map f encodes the following structural morphism in $\mathbf{FVect} \times P$:

$$(N \otimes (N \otimes S \otimes N) \otimes (N \otimes N) \otimes N, n(n^r s n^l)(nn^l)n) \xrightarrow{(f,\leq)} (S, s) \tag{3.52}$$

and arises from the syntactic reduction map (3.53):

$$f = \epsilon_N \otimes 1_S \otimes \epsilon_N \otimes \epsilon_N : N \otimes (N \otimes S \otimes N) \otimes (N \otimes N) \otimes N \to S \tag{3.53}$$

The matrix of f has $dim(N)^2 \times dim(S) \times dim(N)^2 \times dim(N)^2$ columns and $dim(S)$ rows. When applied to the vectors of the meanings of the words we obtain:



(3.54)

Letting $\mathbf{Comedians} = \sum_i c_i \mathbf{n}_i$, $\mathbf{tell} = \sum_{jkl} c_{jkl} \mathbf{n}_j \otimes \mathbf{s}_k \otimes \mathbf{n}_l$, $\mathbf{funny} = \sum_{mn} c_{mn} \mathbf{n}_m \otimes \mathbf{n}_n$, and $\mathbf{jokes} = \sum_o c_o \mathbf{n}_o$, where $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ are orthonormal bases of N and S respectively, we have

$$\begin{aligned}
\overrightarrow{\textit{Comedians tell funny jokes}} &\equiv f(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{funny} \otimes \mathbf{jokes}) && (3.55)\\
&= \sum_{ijklmno} c_i c_{jkl} c_{mn} c_o \langle \mathbf{n}_i \mid \mathbf{n}_j \rangle \mathbf{s}_k \langle \mathbf{n}_l \mid \mathbf{n}_m \rangle \langle \mathbf{n}_n \mid \mathbf{n}_o \rangle && (3.56)\\
&= \sum_k \mathbf{s}_k \sum_{iln} c_i c_{ikl} c_{ln} c_n && (3.57)\\
&\in S && (3.58)
\end{aligned}$$

Hence, both of our examples (3.37) and (3.51) get mapped to a shared meaning space via syntactically guided collapsing of tensor spaces. This is precisely the power of the DisCo model of semantics: *all* well-formed strings of words, i.e. sentences, exist within a *common* semantic space. An implication of this result is that we can use the inner product to compare semantic vectors. This measure is known as a *degree of similarity*, or *cosine measure*, between meanings of words in distributional approaches to semantics [50]. The DisCo model of semantics allows an extension of this measure to meanings of strings of words, such as (3.48) and (3.57).

Definition 3.7 (Degree of Similarity)**.** *Two strings of words $w_1 \cdots w_k$ and $w'_1 \cdots w'_l$ have a degree of similarity, or cosine distance, m – if and only if their pregroup reductions result in the same grammatical type – where*

$$m = \frac{1}{N \times N'} \left\langle f(\mathbf{w}_1 \otimes \cdots \otimes \mathbf{w}_k) \mid f'(\mathbf{w}'_1 \otimes \cdots \otimes \mathbf{w}'_l) \right\rangle \tag{3.59}$$

for

$$N = |f(\mathbf{w}_1 \otimes \cdots \otimes \mathbf{w}_k)| \qquad N' = f'(\mathbf{w}'_1 \otimes \cdots \otimes \mathbf{w}'_l) \tag{3.60}$$

and

$$|\mathbf{v}|^2 = \langle \mathbf{v} \mid \mathbf{v} \rangle \tag{3.61}$$

with f and f' defined as the meaning maps.

3.7 Frobenius Algebra

A recent extension to the field of compositional distributional semantics concerns an account of subject and object relative pronouns within the categorical framework [47]. The difficulty with modelling pronouns in a distributional setting is that contextual analysis fails to provide adequate semantic representations. This is because pronouns tend to appear near a great number of words, and so co-occurrence statistics naturally produce dense general vectors. [47] develops a semantic model of relative pronouns using the general operations of a Frobenius algebra over vector spaces. In this interpretation, relative pronouns play purely structural roles involved in the passing of information between clauses, as well as the combining, copying, and deleting of components of the relative clause.

In the category of finite dimensional vector spaces and linear maps **FVect**, a vector space V with a fixed basis $\{\mathbf{v}_i\}_i$ has a Frobenius algebra over it, explicitly given by:

$$\begin{aligned} &\Delta :: \mathbf{v}_i \mapsto \mathbf{v}_i \otimes \mathbf{v}_i && \iota :: \mathbf{v}_i \mapsto 1 \\ &\mu :: \mathbf{v}_i \otimes \mathbf{v}_j \mapsto \delta_i^j \mathbf{v}_i = \begin{cases} \mathbf{v}_i, & i = j \\ \mathbf{0}, & i \neq j \end{cases} && \zeta :: 1 \mapsto \sum_i \mathbf{v}_i \end{aligned} \tag{3.62}$$

where the Frobenius condition is:

$$(\mu \otimes 1_A) \circ (1_A \otimes \Delta) = \Delta \circ \mu = (1_A \otimes \mu) \circ (\Delta \otimes 1_A) \tag{3.63}$$

Furthermore, Frobenius algebras over vector spaces with orthonormal bases are *commutative* and *special*. A commutative Frobenius algebra satisfies the following two conditions for $\sigma : A \otimes B \to B \otimes A$ (the 'swap' morphism):

$$\sigma \circ \Delta = \Delta \quad \mu \circ \sigma = \mu \tag{3.64}$$

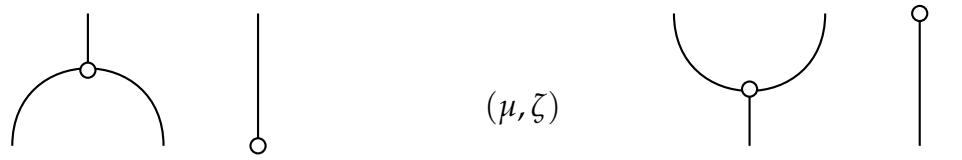
A special Frobenius algebra satisfies the following axiom:

$$\mu \circ \Delta = 1 \tag{3.65}$$

The vector spaces considered in the DisCo model of semantics have fixed orthonormal bases. As such, commutative and special Frobenius algebras exist over them.

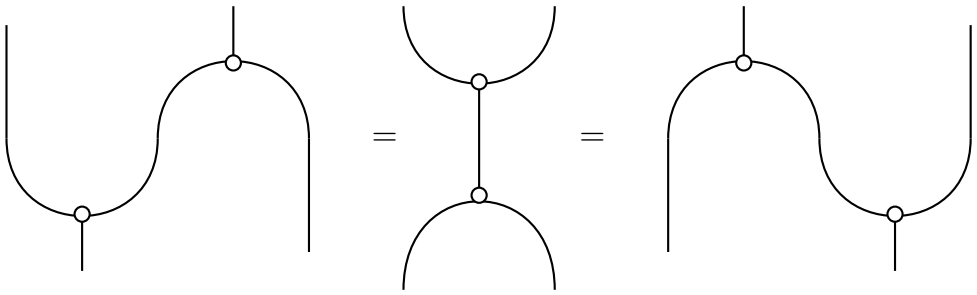
Diagrammatically, we can depict the monoid and comonoid morphisms as:

(Δ, ι) (μ, ζ)



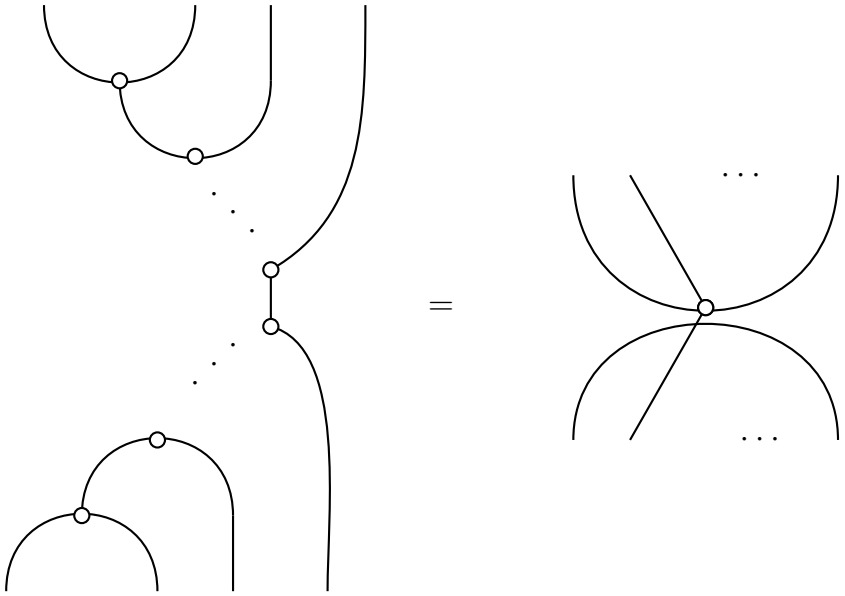
(3.66)

The Frobenius condition is depict as:



(3.67)

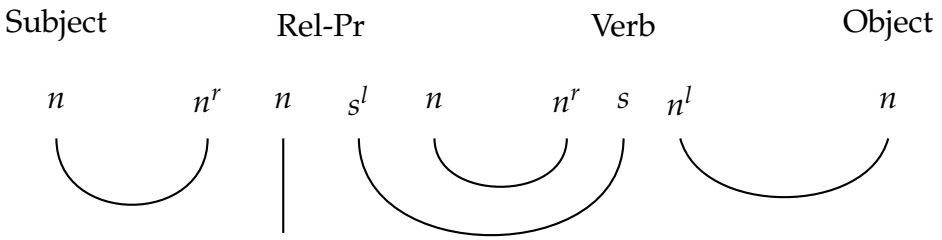
What's more, the definitions of a commutative special Frobenius algebra ensure that any graphical depiction of a Frobenius operation can reduced to a normal form dependent solely on the number of input and output connections of the nodes (independent of their topology). These normal forms can be simplified to 'spiders':



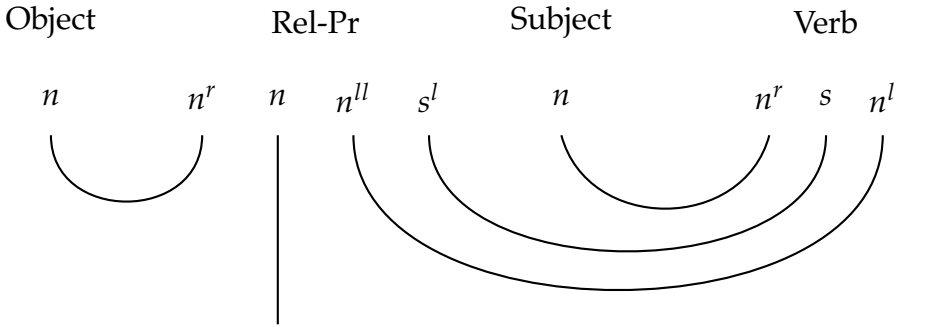
(3.68)

The pregroup types of relative pronouns – e.g. *who(m)*, *which*, *that* – are $n^r n s^l n$

(subject) and $n^r n n^{ll} s^l$ (object). These types result in the following reductions:



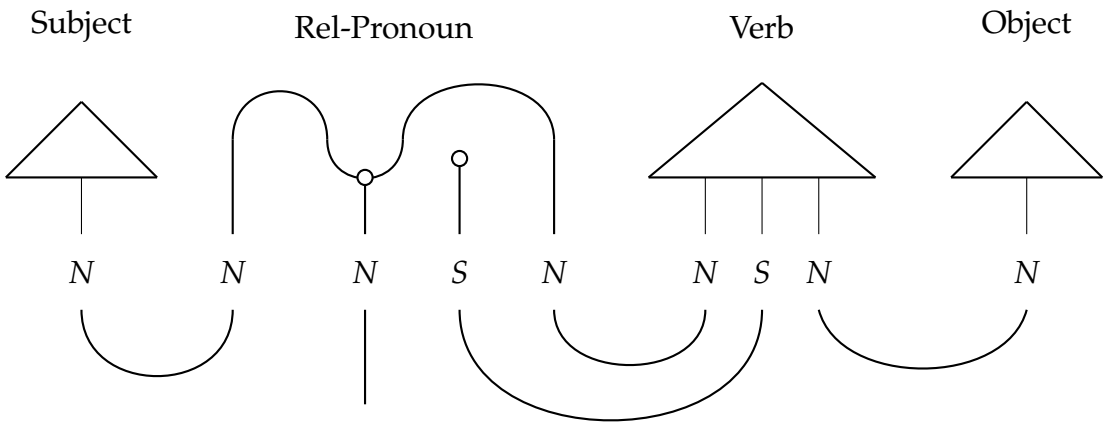
(3.69)



(3.70)

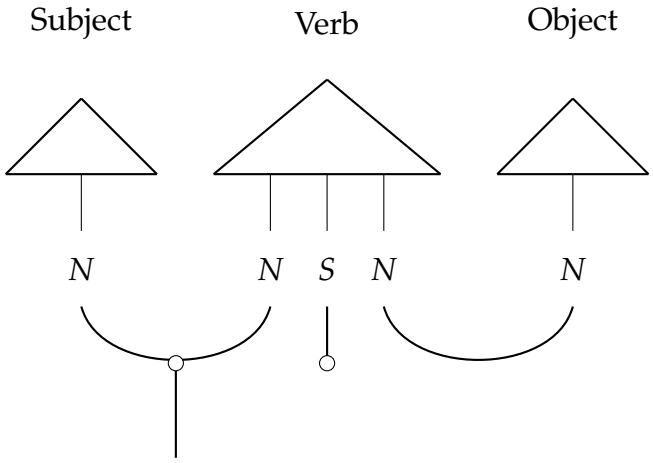
The semantic role that relative pronouns play are revealed in their categorical vector space meaning. Relative pronouns serve to pass information from a clause to a head noun via η maps, combine information via the μ map, and also discard a clause via the ζ map, thus returning a modified noun via 1_N.

The diagram of a meaning vector of a subject relative clause composing with a head noun is:



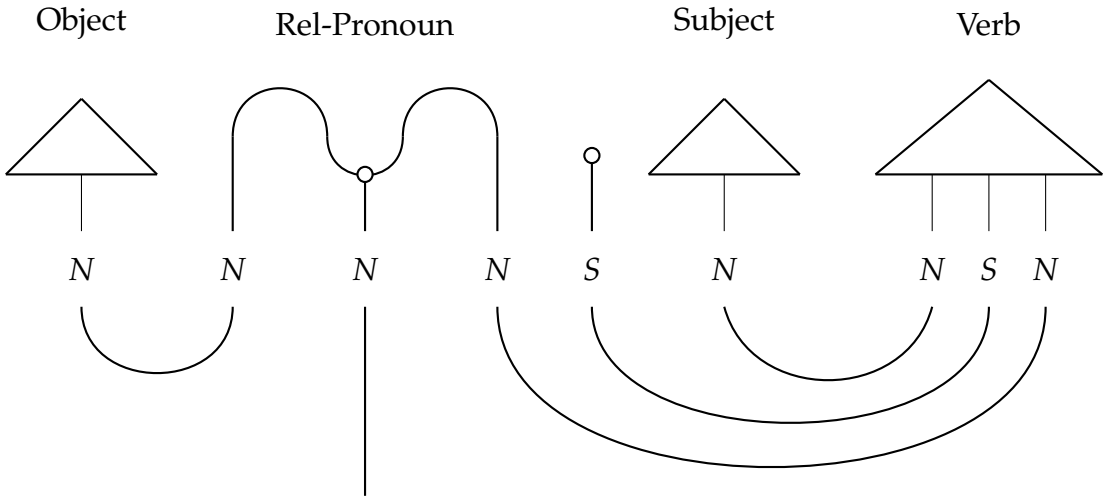
(3.71)

which reduces to:



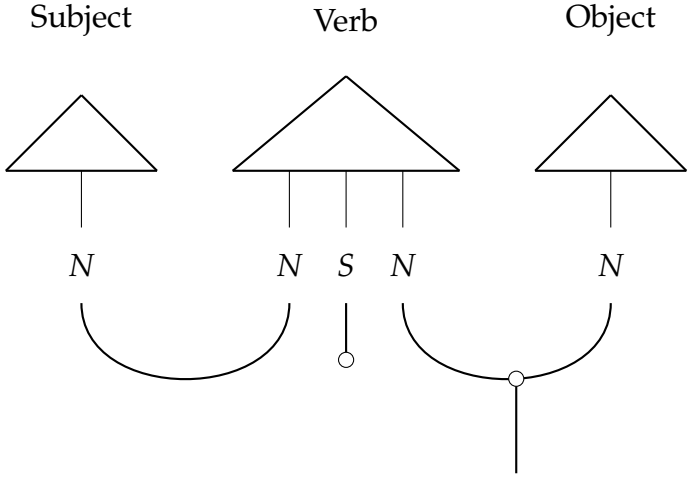
(3.72)

The diagram of a meaning vector of an object relative clause is:



(3.73)

which reduces to:



(3.74)

Example 3.3 (Subject Relative Pronoun)**.** Consider the sentence

$$\textit{Comedians who tell jokes.} \tag{3.75}$$

Letting $\mathbf{Comedians} = \sum_i c_i \mathbf{n}_i$, $\mathbf{tell} = \sum_{jkl} c_{jkl} \mathbf{n}_j \otimes \mathbf{s}_k \otimes \mathbf{n}_l$, and $\mathbf{jokes} = \sum_m c_m \mathbf{n}_m$, where $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ are orthonormal bases of N and S respectively, we have

$$\begin{aligned}
\overrightarrow{\textit{Comedians who tell jokes}} &\equiv f(\mathbf{Comedians} \otimes \textit{who} \otimes \mathbf{tell} \otimes \mathbf{jokes}) && (3.76)\\
&= \mu_N \otimes \iota_S \otimes \epsilon_N(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) && (3.77)\\
&= \mu_N \otimes \iota_S(\mathbf{Comedians} \otimes (\sum_{jkl} c_{jkl} c_l \mathbf{n}_j \otimes \mathbf{s}_k)) && (3.78)\\
&= \mu_N(\mathbf{Comedians} \otimes (\sum_{jkl} c_{jkl} c_l \mathbf{n}_j)) && (3.79)\\
&= \sum_{ij} \delta_i^j c_i \mathbf{n}_i \sum_{kl} c_{jkl} c_l && (3.80)\\
&= \sum_i c_i \mathbf{n}_i \sum_{kl} c_{ikl} c_l && (3.81)\\
&\in N && (3.82)
\end{aligned}$$

(3.81) gives us an intuitive distributional representation of (3.75): the sum of the subject individuals (i.e. *comedians*) weighted by the degree to which they have acted on the object individuals (i.e. *jokes*) via the verb (i.e. *tell*).

Example 3.4 (Object Relative Pronoun)**.** Consider the sentence

$$\textit{Jokes which comedians tell.} \tag{3.83}$$

Letting $\mathbf{Jokes} = \sum_i c_i \mathbf{n}_i$, $\mathbf{comedians} = \sum_j c_j \mathbf{n}_j$, and $\mathbf{tell} = \sum_{klm} c_{klm} \mathbf{n}_k \otimes \mathbf{s}_l \otimes \mathbf{n}_m$, where $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ are orthonormal bases of N and S respectively, we have

$$\begin{aligned}
\overrightarrow{\textit{Jokes which comedians tell}} &\equiv f(\mathbf{Jokes} \otimes \textit{which} \otimes \mathbf{comedians} \otimes \mathbf{tell}) && (3.84)\\
&= \epsilon_N \otimes \iota_S \otimes \mu_N(\mathbf{comedians} \otimes \mathbf{tell} \otimes \mathbf{Jokes}) && (3.85)\\
&= \iota_S \otimes \mu_N((\sum_{jlm} c_j c_{jlm} \mathbf{s}_l \otimes \mathbf{n}_m) \otimes \mathbf{Jokes}) && (3.86)\\
&= \mu_N((\sum_{jlm} c_j c_{jlm} \mathbf{n}_m) \otimes \mathbf{Jokes}) && (3.87)\\
&= \sum_{im} \delta_i^m c_i \mathbf{n}_i \sum_{jl} c_j c_{jlm} && (3.88)\\
&= \sum_i c_i \mathbf{n}_i \sum_{jl} c_j c_{jli} && (3.89)\\
&\in N && (3.90)
\end{aligned}$$

The question of whether computers can think is like the question of whether submarines can swim.

— Edsger Dijkstra

4 Compositional Distributional Cognition

[52] presents a view of cognition that incorporates two distinct but related levels of formal description: "the continuous, numerical lower-level description of the brain", characterized in terms of a connectionist network, and "the discrete, structural higher-level description of the mind", characterized in terms of symbolic rules [41]. The design of the *Integrated Connectionist/Symbolic Architecture* (ICS) – a unified connectionist and symbolic cognitive architecture – is a novel approach to the computational modelling of the mind with much promise. However, the tensor product representations used to codify the isomorphism between connectionist and symbolic representations reveal a number of shortcomings. Firstly, the representational space of a concept grows in size as more elements are added to the compound. Secondly, only symbolic representations with the same underlying structure can be compared.

As we have seen, [12] introduces a mathematical framework for a unification of the distributional theory of meaning in terms of vector space models and the compositional theory of grammatical types. The compositional distributional (DisCo) model of semantics utilises grammar in order to use composite spaces without increase in size of the resulting meaning space and allows composite concepts to be directly compared with their constituents, as well as the meaning of sentences of varying length and structure to be compared. The similarities between the linguistic problem motivating the DisCo model of semantics and the drawbacks of ICS suggest that the interaction of quantum mechanics (QM) and cognitive science is a fruitful area of research for artificial intelligence (AI). In our case, we aim to take advantage of the DisCo architecture in order to harness the potential of ICS.

ICS		DisCo
$\mathbf{f} \in V_F$	$\mathtt{f}$	$w \in W$
$\mathbf{r}_x \in V_R^{\otimes \lvert x \rvert}$	r_x	$p \in P$
$\mathbf{f}_i \otimes \mathbf{r}_{x_i}$	$\mathtt{f}_i / r_{x_i}$	$(f, \leq) : (V, p) \to (W, q)$
$\mathbf{s} = \sum_i \mathbf{f}_i \otimes \mathbf{r}_{x_i} \in \mathcal{S}^*$	$\mathbf{s} = \{\mathtt{f}_i / r_{x_i}\}$	$(W_1 \otimes \cdots \otimes W_n, p_1 \cdots p_n) \xrightarrow{(f,\leq)} (S, s)$

Table 4.1: Space of descriptions in ICS and DisCo

Here, we present the DisCo model of mind: compositional distributional cognition (DisCog). Specifically, we build a concrete QM model of ICS based on the DisCo framework, putting together all the basic components of ICS – i.e. representation, processing, harmony, and harmonic grammar – with a view to exploring the results of a more sophisticated mathematical analysis. This work provides the main deck of the bridge – linking cognitive science and quantum mechanics – which we ultimately aim to build.

4.1 Unpacking Grammar

DisCog hinges on adapting the approach layed out in [12] to allow grammatical types describing ICS-like *roles* to be easily represented in a vector space.

4.1.1 ICS versus DisCo

Consider the sentence

$$\textit{Comedians tell jokes.} \tag{4.1}$$

At the process level in ICS, (4.1) corresponds to (4.3):

$$\mathbf{s} = \mathbf{Comedians} \otimes \mathbf{r}_0 + (\mathbf{tell} \otimes \mathbf{r}_0 + \mathbf{jokes} \otimes \mathbf{r}_1) \otimes \mathbf{r}_1 \tag{4.2}$$

$$= \mathbf{Comedians} \otimes \mathbf{r}_0 + \mathbf{tell} \otimes \mathbf{r}_{01} + \mathbf{jokes} \otimes \mathbf{r}_{11} \tag{4.3}$$

In the DisCo model of semantics, (4.1) corresponds to (4.7):

$$\overrightarrow{\textit{Comedians tell jokes}} \equiv f(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) \tag{4.4}$$

$$= \epsilon_N \otimes 1_S \otimes \epsilon_N(\mathbf{Comedians} \otimes \mathbf{tell} \otimes \mathbf{jokes}) \tag{4.5}$$

$$= \sum_{ijklm} c_i c_{jkl} c_m \langle \mathbf{n}_i \mid \mathbf{n}_j \rangle \mathbf{s}_k \langle \mathbf{n}_l \mid \mathbf{n}_m \rangle \tag{4.6}$$

$$= \sum_k \mathbf{s}_k \sum_{il} c_i c_{ikl} c_l \tag{4.7}$$

The tensors $(\mathbf{Comedians} \otimes \mathbf{r}_0)$, $(\mathbf{tell} \otimes \mathbf{r}_{01})$, and $(\mathbf{jokes} \otimes \mathbf{r}_{11})$ in (4.3) are pure tensors and thus can be interpreted as pairs of vectors, i.e. $(\mathbf{Comedians}, \mathbf{r}_0)$, $(\mathbf{tell}, \mathbf{r}_{01})$, and $(\mathbf{jokes}, \mathbf{r}_{11})$. These are pairs of a meaning of a word and its grammatical role and almost identical to the pairs considered in the DISCO approach, i.e. those of a meaning space of each word. A minor notational difference between the representations is that, in ICS, the grammatical role $\mathbf{r}_x$ is a genuine vector, whereas in the DISCO model p remains a grammatical type. A major qualitative difference between the two approaches, however, is that ICS treats all words as simple vectors ($\in V_F$), whereas in the DISCO model the vector of a verb itself lives in a tensor space, e.g. $\mathbf{tell} \in N \otimes S \otimes N$, as do adjectives and other words with compound types. Seeing that ICS realizes filler/role bindings via tensor product representations, this variance between the respective models is not so extreme. Still, intuition tells us that certain words, such as adjectives and verbs, have fundamentally different structure to nouns and therefore warrant a more sophisticated treatment.

What's important to remember is that, in ICS, information in the mind/brain is processed by widely distributed connection patterns, i.e. weight matrices:

$$\mathcal{P}(\mathbf{s}) = \mathbb{W} \cdot \mathbf{s} \tag{4.8}$$

$$= (\mathbb{I} \otimes \underline{\mathbb{W}}) \cdot \mathbf{s} \tag{4.9}$$

The reason why ICS provides a means of constructing simple networks which compute arbitrarily complex symbolic (recursive) functions is due to the fact that $\mathbb{W}$ possesses certain global structure: $\mathbb{W}$ is a certain product and sum of fundamental matrices, e.g. $\mathbb{W}_{\text{ex0}}$, $\mathbb{W}_{\text{ex1}}$, $\mathbb{W}_{\text{cons0}}$, and $\mathbb{W}_{\text{cons1}}$ in the case of a binary tree. These matrices are precisely the kind of linear maps ICS employs to compute symbolic representations. So (4.2) can be interpreted as:

$$\mathbb{W}_{\text{cons0}} \cdot \mathbf{Comedians} + \mathbb{W}_{\text{cons1}} \cdot (\mathbb{W}_{\text{cons0}} \cdot \mathbf{tell} + \mathbb{W}_{\text{cons1}} \cdot \mathbf{jokes}) \tag{4.10}$$

(4.10) makes clear how compositionality is achieved in ICS: fundamental matrices, e.g. $\mathbb{W}_{\text{cons}x}$, encode compositional rules.

As we have already noted, ICS represents all atomic fillers as simple vectors. As such, learning compositionality amounts to learning the entries of fundamental matrices which are general to a certain cognitive task. This is achieved via application of a learning procedure to the underlying connectionist network. In the DISCO model, analysis of word co-occurrence statistics gives us vectors which live in tensor spaces for words with compound grammatical types (e.g. verbs, adjective, etc.). That being so, the DISCO model makes no difference in its approach

to learning compositionality from conventional methods used in distributional semantics to learn simple word vectors, i.e. linguistic fillers in ICS. Conceptually, the contrast between the two approaches boils down to a choice between learning algorithms: our implicit claim is that the former's neural network model is biologically inspired and thus closer to cognitive learning. While a discussion on the pros and cons of the respective learning algorithms is beyond the scope of this work, the analysis thus far helps to shed light on how to endow grammatical types characterizing ICS-like roles with vector space structure in order to formalize a DISCO adaption of ICS which fundamentally rests on connectionist principles.

Examining (4.6), we recognize that c_{jkl}, $\mathbf{n}_j$, and $\mathbf{n}_l$ encode how the *action* component $\mathbf{s}_k$ of a transitive verb interacts, or *composes*, with its subject and object, or *arguments*, respectively, to produce a sentence $\mathbf{s} \in S$. Therefore, when shifting from the DISCO model of semantics to ICS, we must faithfully represent these compositional rules in a general way and in doing so extract a simple connectionist representation of a transitive verb as well as other compound grammatical types. While it may be tempting to separate the composite state **tell** – which is a linear combination of many separable tensors – we must be mindful of the fact that entangled states are necessary to allow the flow of information between different subsystems [28]. More concretely, to compute the meaning of the whole sentence, the meaning of the verb will need to *interact* with the meaning of both the *subject* and the *object*, so it cannot be decomposed into three disconnected entities:

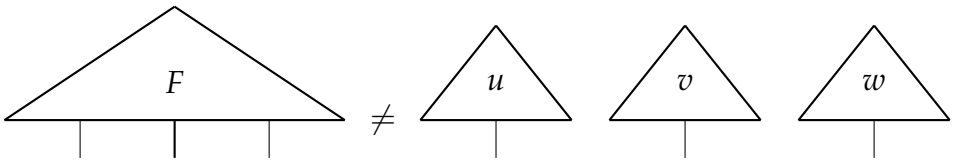


Figure 4.1: Entanglement of a composite state

If all relational words were represented by separable tensors, the normalized meaning of all sentences with **verb** as the verb would be identical and equivalent to the S component of **verb**. Clearly, this is an undesirable property since *Comedians tell jokes* and *Reporters tell facts* should not have the same meaning.[1]

The inherent difference between compound types and atomic types suggests that our adaption of ICS must treat words like verbs in a distinct manner to nouns.

[1]Euclidian distance also detects differences in magnitude and therefore fairs marginally better than the usual cosine measure, however, this metric has been shown to be inadequate for distributional models of meaning.

Given that nouns are represented as simple vectors in both ICS and the DISCO model of semantics, it seems appropriate to preserve their interpretation as atomic (linguistic) fillers ($\in V_F$). As further motivation, intuition tells us that nouns are the building blocks of language. Without nouns, we have no way of describing objects and therefore no method for ascribing properties to the fabric of the universe. For this reason, we are inclined to model compound types – which correspond to tensors in **FVect** via the linear map f – as roles.

This definition is, in fact, very natural. Realizations of compound types are essentially functional arguments that encode entanglement. As such, they are the perfect candidates for occupying roles which govern compositional rules in DISCOG.[2] A neat way to think about realizations of compound types is with reference to function pointers in programming languages. Instead of holding data values, function pointers point to executable code within memory. When dereferenced, a function pointer invokes the function it points to and passes along arguments just like a routine function call. In our case, realizations of the same compound type point to some canonical function – equating to the logical aspect of compositional semantics – however, pass *unique* arguments specified by entanglement encoding – coinciding with the distributional approach to semantics.

4.1.2 Tensors as Roles

Let us reiterate that the point of modelling compound types as roles is to derive a simple connectionist representation of the DISCO model of semantics. Doing so will give us a semantic representation that we can then transfer over to our adaptation of ICS, DISCOG. The hope is that our new computational model of mind will solve the basic representational problems with ICS and, at the same time, recover the fundamental connectionist notion of harmony. As a reminder, the general cognitive representations considered in ICS are of the form:

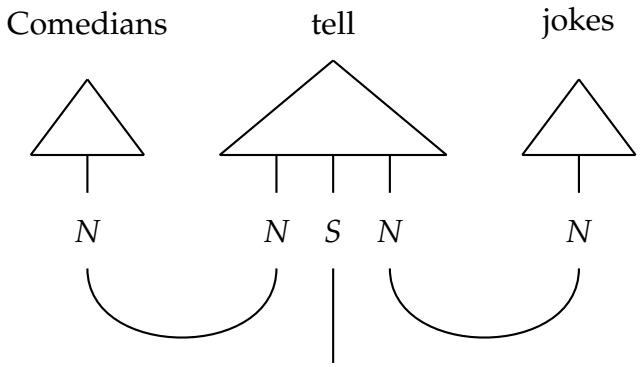
$$\mathbb{W} \cdot \mathbf{f} \tag{4.11}$$

(4.11) denotes a filler/role binding $\mathtt{f}/r$, where $\mathbb{W}$ is the realization of a role r and $\mathbf{f}$ is the realization of a filler $\mathtt{f}$.

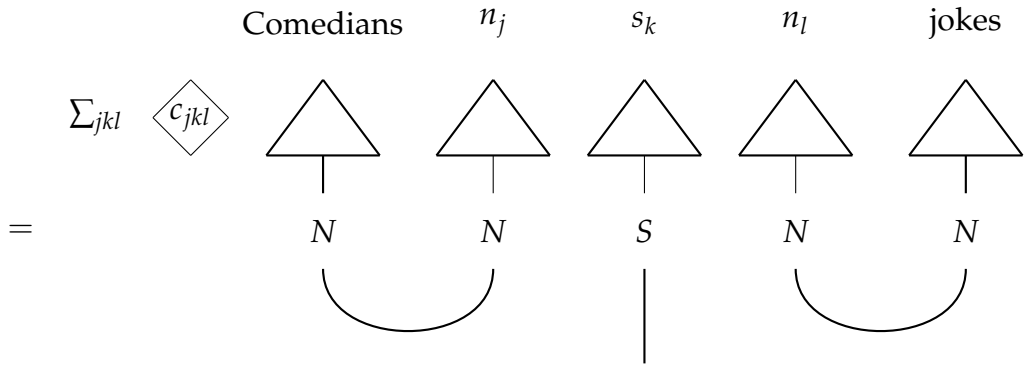
Returning to our example (4.1), we now dissect the DISCO model of semantics formulation (4.5) in order to extract a filler/role representation of meaning. (Henceforth, we readily use the graphical notation which simplifies meaning computations to a great extent and exposes information flow in a manner ideal for a

[2] The hint is in the name!

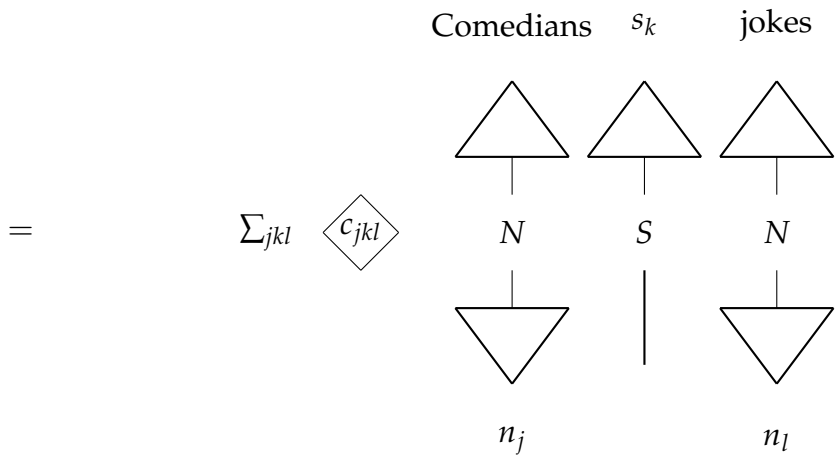
rigorous, yet high level analysis.) In particular, we reduce (4.5) to (4.17) using the diagrammatic calculus:



(4.12)

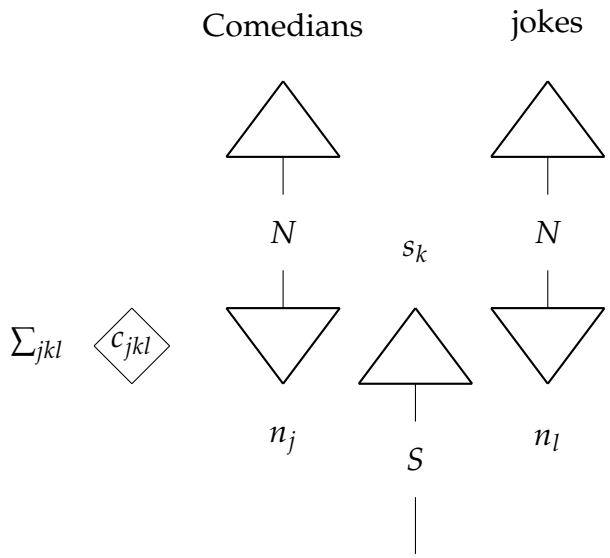


(4.13)



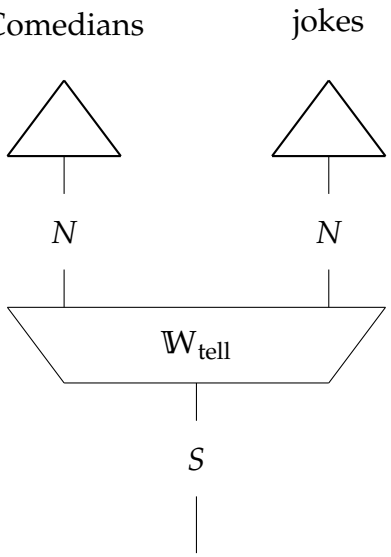
(4.14)

$=$



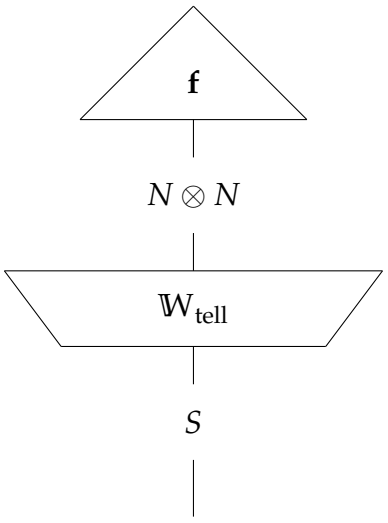
(4.15)

$=$



(4.16)

$\equiv$



(4.17)

Let us walk through this derivation. (4.12) is simply the application of the meaning map f to the vectors of the meanings of the words: **Comedians**, **tell**, and **jokes**. (4.13) then follows from unpacking the definition of **tell**. (4.14) is the result of the 'swing' graphical rewrite rule. Next, (4.15) amounts to 'sliding' the summation and all of its components down the page. Since the topology of the structure is preserved, (4.15) is equivalent to (4.14). We arrive at (4.16) via grouping together the individual parts of the summation (on bottom) in (4.15) and defining a matrix $\mathbb{W}_{\text{tell}}$ such that the equality holds (more on this later). Finally, we group together the fillers (on top) in (4.16) to obtain (4.17) and define:

$$\mathbf{f} \equiv \mathbf{Comedians} \otimes \mathbf{jokes} \tag{4.18}$$

(4.17) gives us a nice representation derived from the DISCO model of semantics that mirrors the filler/role perspective at the heart of ICS. Symbolically, we can depict these pairings of fillers and roles as a dependency tree:

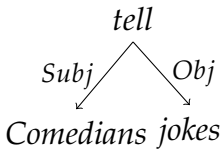
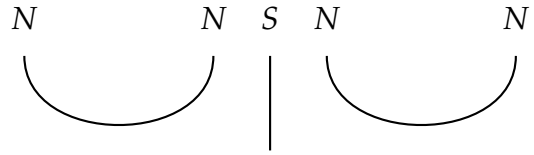


Figure 4.2: Example dependency tree

What is especially attractive about (4.17) is the immediate visualization of the process by which connectionist properties percolate up[3] to the symbolic level. Much in the same way that system calls interface between a kernel-level operating system and user-level applications in computer architecture, weight matrices that are realizations of roles connect the two layers of abstraction in our cognitive architecture: the lower-level underlying connectionist network, i.e. central nervous system, and the higher-level symbol manipulating computer, i.e. mind. Like an operating system kernel, the connectionist network has privileged access to the hardware of the brain with its own kind of CPU, memory, and I/O. Akin to user-level applications, the mind – a rule-governed serial device – has its own set of registers and working memory. In this sense, the weight matrices that realize roles in DISCOG form the context boundary, or API, separating two very different modes of cognition.

[3]In our graphical notation, connectionist properties percolate down to the symbolic level since by convention information flows from top to bottom in the diagrammatic calculus of monoidal categories.

It still remains to describe the internal structure of $\mathbb{W}_{\text{tell}}$ to establish the equality between (4.15) and (4.16). $\mathbb{W}_{\text{tell}}$ is very closely related to the matrix of the linear map f which takes us from meanings of words to meaning of a sentence. As we have seen before, the matrix of f for our running example – *Comedians tell jokes* – has $dim(N)^2 \times dim(S) \times dim(N)^2$ columns and $dim(S)$ rows as shown in (4.19):



(4.19)

The difference between $\mathbb{W}_{\text{tell}}$ and the matrix of f boils down to our decision to partition tensors and (simple) vectors into *roles* and *fillers* respectively – a procedure that will ultimately prove to be crucial for distilling a simple ICS-like connectionist representation of compositionality inherent within the DisCo model. We now illustrate the passage from f to $\mathbb{W}_{\text{tell}}$.

Recall that $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ are orthonormal bases of N and S respectively. By the rules of matrix algebra, we have

$$[\sum_k \mathbf{s}_k \sum_{jl} c_j c_{jkl} c_l]_n = \sum_{jl} c_j c_{jnl} c_l \tag{4.20}$$

$$= \sum_{jl} c_{jnl} c_j c_l \tag{4.21}$$

where $[\mathbf{v}]_n$ is the n^{th} component of a vector $\mathbf{v}$. Setting $c_{abc} = c^{\mathbf{word}}_{a:b:c}$ and $d_V = dim(V)$, we wan rewrite (4.21) as (4.23):

$$\left[c^{\mathbf{tell}}_{1:n:1} \quad \cdots \quad c^{\mathbf{tell}}_{j:n:l} \quad \cdots \quad c^{\mathbf{tell}}_{d_N:n:d_N}\right] \cdot \left(\begin{bmatrix} c^{\mathbf{Comedians}}_1 \\ \vdots \\ c^{\mathbf{Comedians}}_j \\ \vdots \\ c^{\mathbf{Comedians}}_{d_N} \end{bmatrix} \otimes \begin{bmatrix} c^{\mathbf{jokes}}_1 \\ \vdots \\ c^{\mathbf{jokes}}_l \\ \vdots \\ c^{\mathbf{jokes}}_{d_N} \end{bmatrix} \right) \tag{4.22}$$

$$\equiv [\mathbb{W}_{\text{tell}}]_n \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) \tag{4.23}$$

where $[\mathbb{W}_{\text{tell}}]_n$ is the n^{th} row of the matrix $\mathbb{W}_{\text{tell}}$ with $dim(N) \times dim(N)$ columns and $dim(S)$ rows. Let $[\mathbb{W}_{\text{tell}}]^c_r$ denote the $1 \times d_N$-dimensional submatrix occupying block entry row r column c of $\mathbb{W}_{\text{tell}}$, i.e.

$$[\mathbb{W}_{\text{tell}}]_n = \left[[\mathbb{W}_{\text{tell}}]^1_n \quad [\mathbb{W}_{\text{tell}}]^2_n \quad \cdots \quad [\mathbb{W}_{\text{tell}}]^{d_N}_n\right] \tag{4.24}$$

Furthermore, let $[\mathbf{tell}]_{(m,m+d_N]}$ denote the i^{th} d_N-dimensional subvector of **tell** where $m = (i-1) \times d_N$ with $i = [1, d_N d_S]$, i.e.

$$\mathbf{tell} = \begin{bmatrix} [\mathbf{tell}]_{(0,d_N]} \\ [\mathbf{tell}]_{(d_N,2d_N]} \\ \vdots \\ [\mathbf{tell}]_{[(m,m+d_N]} \\ \vdots \\ [\mathbf{tell}]_{(d_N^2 d_S - d_N, d_N^2 d_S]} \end{bmatrix} \tag{4.25}$$

where

$$[\mathbf{tell}]_{(m,m+d_N]} = \begin{bmatrix} c^{\mathbf{tell}}_{j:k:1} & \cdots & c^{\mathbf{tell}}_{j:k:d_N} \end{bmatrix}^{\mathrm{T}} \tag{4.26}$$

for $j = \lceil \frac{i}{d_S} \rceil$ and $k = i - [(j-1) \times d_S]$. We can construct $\mathbb{W}_{\text{tell}}$ from $\mathbf{tell} \in N \otimes S \otimes N$ via the equality:

$$[\mathbf{tell}]_{(m,m+d_N]} = [\mathbb{W}_{\text{tell}}]_r^{c\mathrm{T}} \tag{4.27}$$

for $i = [(c-1) \times d_S] + r$. (In other words, $[\mathbb{W}_{\text{tell}}]_r^c$ is the transpose of the i^{th} d_N-dimensional subvector of **tell** that fills block entry row r column c of $\mathbb{W}_{\text{tell}}$ for $i = [(c-1) \times d_S] + r$.) Therefore, we have

$$\mathbf{tell} = \begin{bmatrix} [\mathbb{W}_{\text{tell}}]_1^{1\mathrm{T}} \\ [\mathbb{W}_{\text{tell}}]_2^{1\mathrm{T}} \\ \vdots \\ [\mathbb{W}_{\text{tell}}]_r^{c\mathrm{T}} \\ \vdots \\ [\mathbb{W}_{\text{tell}}]_{d_S}^{d_N\mathrm{T}} \end{bmatrix} \tag{4.28}$$

where

$$\mathbb{W}_{\text{tell}} = \begin{bmatrix} [\mathbb{W}_{\text{tell}}]_1^1 & [\mathbb{W}_{\text{tell}}]_1^2 & \cdots & [\mathbb{W}_{\text{tell}}]_1^{d_N} \\ [\mathbb{W}_{\text{tell}}]_2^1 & [\mathbb{W}_{\text{tell}}]_2^2 & \cdots & [\mathbb{W}_{\text{tell}}]_2^{d_N} \\ \vdots & \vdots & \ddots & \vdots \\ [\mathbb{W}_{\text{tell}}]_{d_S}^1 & [\mathbb{W}_{\text{tell}}]_{d_S}^2 & \cdots & [\mathbb{W}_{\text{tell}}]_{d_S}^{d_N} \end{bmatrix} \tag{4.29}$$

$$= \begin{bmatrix} [\mathbb{W}_{\text{tell}}]_1 \\ \vdots \\ [\mathbb{W}_{\text{tell}}]_{d_S} \end{bmatrix} \tag{4.30}$$

Hence, our final semantic representation for *Comedians tell jokes* is realized by (4.32):

$$\overrightarrow{\textit{Comedians tell jokes}} = \mathbb{W}_{\text{tell}} \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) \tag{4.31}$$

$$\equiv \mathbb{W}_{\text{tell}} \cdot \mathbf{f} \tag{4.32}$$

$$\in S \tag{4.33}$$

where $\mathbb{W}_{\text{tell}}$ faithfully represents the compositional rules encoded in **tell** and **f** is the tensor of atomic (linguistic) fillers, i.e. **Comedians** and **jokes**.

In sum, we have arrived at a simple connectionist representation of compositionality derived from the DISCO model that is identical to the general cognitive representations considered in ICS. As such, (4.17) offers some insight concerning how realizations of well-formed symbols comprised of pairings of fillers and roles can be mapped to a *shared* meaning space. This kind of approach will help us solve the shortcomings encountered in ICS because realizations of symbolic representations no longer need live in an unbounded representational space $\mathcal{S}^*$. What's more, symbolic representations with different underlying structures can now be compared. It is important to note that from the standpoint of ICS and DISCOG, c_{jkl} in (4.17) are initially unknown.[4] These values – which encode entanglement – will be learned by the underlying connectionist network via backpropagation. Nonetheless, a presentation of the formal equivalence between representations in the DISCO model of semantics and DISCOG remains necessary.

4.2 An Alternative Connectionist Realization

Here, we formalize the ideas discussed in the previous section. First, we give a connectionist account of the (ϵ, ι, and μ) maps required for information flow in DISCOG.[5] We then present an alternative connectionist realization taking us from activation values to symbols that not only resolves the representational issues encountered in ICS, but also offers a more sophisticated notion of compositionality. Our new realization notably differs from ICS in that our definition is recursive on the matrix-vector multiplication operation. Thereafter, we work through a number of linguistic examples which demonstrate the validity of our model and provide explanatory value.

[4] A reasonable strategy could be to initialize c_{jkl} with values learned from co-occurrence statistics, as opposed to straightforward random initialization. However, such an approach would *seem* to undermine the realism of our model of mind. Having said that, there is much debate in philosophy of language regarding the extent to which language is innate. Hatching up an answer to this question is a task for neuroscientists, biologists, linguists, philosophers, and the like.

[5] In the following sections, we assume the notation $d_V = dim(V)$.

4.2.1 Information Flow

To enable compositional distributional information flow – which we believe to be an elegant form of spreading activation – in the underlying connectionist network of our cognitive architecture, we must offer a formal equivalence between the reduction maps used in the DISCO model of semantics and the mechanisms implemented in DISCOG. This entails transforming the ϵ, ι, and μ maps into *elementary* connectionist operations – linear associators. Once we have this basic instruction set for linear processing in our network, we can then begin to explore the passage from activation values to symbolic structures. In addition to proving the equivalence between our fundamental connectionist operations and the pertinent compact closure/Frobenius algebra maps, we provide supplementary computational algorithms to construct our linear associators.

Proposition 4.1 (ϵ Map in **FVect** as Matrix-Vector Multiplication)**.** *For a tensor*

$$\mathbf{t} = \mathbf{v} \otimes \mathbf{r} \otimes \mathbf{w} \tag{4.34}$$

$$\in V \otimes (V \otimes S \otimes W) \otimes W \tag{4.35}$$

with

$$\mathbf{r} = (\sum_{ijk} c_{ijk} \mathbf{v}_i \otimes \mathbf{s}_j \otimes \mathbf{w}_k) \tag{4.36}$$

$$\in V \otimes S \otimes W \tag{4.37}$$

and $\{\mathbf{v}_i\}_i \equiv V$, $\{\mathbf{s}_j\}_j \equiv S$, *and* $\{\mathbf{w}_k\}_k \equiv W$, *the* ϵ *map is defined by*

$$\epsilon_V \otimes 1_S \otimes \epsilon_W :: \mathbf{t} \mapsto \mathbb{W} \cdot (\mathbf{v} \otimes \mathbf{w}) \tag{4.38}$$

where

$$\mathbb{W} = \begin{bmatrix} [\mathbb{W}]_1^1 & [\mathbb{W}]_1^2 & \cdots & [\mathbb{W}]_1^{d_V} \\ [\mathbb{W}]_2^1 & [\mathbb{W}]_2^2 & \cdots & [\mathbb{W}]_2^{d_V} \\ \vdots & \vdots & \ddots & \vdots \\ [\mathbb{W}]_{d_S}^1 & [\mathbb{W}]_{d_S}^2 & \cdots & [\mathbb{W}]_{d_S}^{d_V} \end{bmatrix} \tag{4.39}$$

and

$$\mathbf{r} = \begin{bmatrix} [\mathbb{W}]_1^{1\mathrm{T}} \\ [\mathbb{W}]_2^{1\mathrm{T}} \\ \vdots \\ [\mathbb{W}]_n^{m\mathrm{T}} \\ \vdots \\ [\mathbb{W}]_{d_S}^{d_V\mathrm{T}} \end{bmatrix} \tag{4.40}$$

where $[\mathbb{W}]_n^m$ *is the transpose of the* i^{th} d_W*-dimensional subvector of* $\mathbf{r}$ *that fills block entry row* n *column* m *of* $\mathbb{W}$ *for* $i = [(m-1) \times d_S] + n$.

Proof. (4.38) follows from (4.20) $\rightarrow$ (4.30). (We can easily construct $\mathbb{W}$ by looping through d_W-dimensional chunks of $\mathbf{r}$ and inserting the transpose of each chunk in the next available slot in $\mathbb{W}$, where we index bottom-down right-to-left.)

□

Algorithm 1 Constructing ϵ in **FVect**

Require: $\mathbf{r}$ is a tensor (i.e. multi-dimensional array) in $V \otimes S \otimes W$

```
function ϵ(r)
    d_V ← LENGTH(r)
    d_S ← LENGTH(r[1])
    d_W ← LENGTH(r[1][1])
    n ← d_S                                          ▷ n: # of rows
    m ← d_V                                          ▷ m: # of block columns
    W ← [[[⊥] × d_W for 1 to m] for 1 to n]           ▷ ⊥: empty
    v ← 1
    s ← 1
    for i ← 1 to m do                                ▷ i: right-to-left
        for j ← 1 to n do                            ▷ j: bottom-down
            W[j][i] ← TRANSPOSE(r[v][s])
            s ← s + 1
            if s > d_S then
                v ← v + 1
                s ← 1
    return W                                         ▷ W: reduce matrix
```

Corollary 4.1 (Recursive ϵ Map Application in **FVect**). *For a rank-*$(m+1+n)$ *tensor* $\mathbf{r}$*, the recursive application of m-left and n-right* ϵ *map reductions via a left-sided tensor product argument* $(\mathbf{p}_1 \otimes \cdots \otimes \mathbf{p}_m)$ *and a right-sided tensor product argument* $(\mathbf{q}_1 \otimes \cdots \otimes \mathbf{q}_n)$*, i.e.*

$$(\mathbf{p}_1 \otimes \cdots \otimes \mathbf{p}_m) \otimes \mathbf{r} \otimes (\mathbf{q}_1 \otimes \cdots \otimes \mathbf{q}_n) \tag{4.41}$$

is defined by

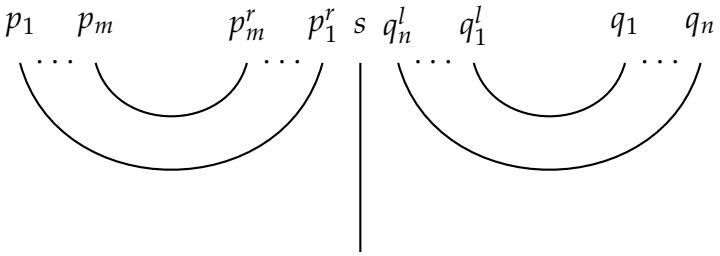
$$\mathbb{W} \cdot (\mathbf{p}_m \otimes \cdots \otimes \mathbf{p}_1 \otimes \mathbf{q}_n \otimes \cdots \otimes \mathbf{q}_1) \tag{4.42}$$

where

$$\mathbb{W} = \epsilon(\mathbf{r}) \tag{4.43}$$

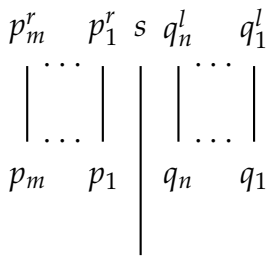
is our reduce matrix.

Proof. Recall that the underlinks representing type reductions in pregroup grammars are exactly the 'cups' of a compact closed structure. Framing our recursive ϵ map reductions in the diagrammatic calculus of pregroup analysis, we have



(4.44)

Applying $m + n$ 'swing' rewrite rules and remembering that the ϵ map takes us to $\mathbb{R}$ in **FVect** (and 1 in P), we obtain



(4.45)

$p_m^r \cdots p_1^r q_n^l \cdots q_1^l$ (on top) corresponds to the columns of $\mathbb{W}$ (i.e. entanglement values) and s corresponds to the rows of $\mathbb{W}$ (i.e. output). Likewise, $p_m \cdots p_1 q_n \cdots q_1$ (on bottom) corresponds to the well-oriented tensor argument (i.e. filler values) that $\mathbb{W}$ requires to ensure the correct connections between components in our matrix-vector multiplication reformulation of the ϵ map. Hence, we arrive at

$$\mathbb{W} \cdot (\mathbf{p}_m \otimes \cdots \otimes \mathbf{p}_1 \otimes \mathbf{q}_n \otimes \cdots \otimes \mathbf{q}_1) \tag{4.46}$$

where $\mathbb{W} = \epsilon(\mathbf{r})$ is our *reduce* matrix. The structure of $\mathbb{W}$ paired with the reverse tensor product of the left- and right-sided arguments preserves the correct topology of recursive ϵ map reductions. (Given the ϵ maps carry out most of the work in computing symbolic meanings from constituent meanings, this result should prove to be rather useful.)

□

Proposition 4.2 (ι Map in **FVect** as Matrix-Vector Multiplication)**.** *For a tensor*

$$\mathbf{t} = \sum_{ijk} c_{ijk} \mathbf{v}_i \otimes \mathbf{s}_j \otimes \mathbf{w}_k \tag{4.47}$$

$$\in V \otimes S \otimes W \tag{4.48}$$

with $\{\mathbf{v}_i\}_i \equiv V$, $\{\mathbf{s}_j\}_j \equiv S$, *and* $\{\mathbf{w}_k\}_k \equiv W$, *the* ι *map is defined by*

$$1_V \otimes \iota_S \otimes 1_W :: \mathbf{t} \mapsto \mathbb{W} \cdot \mathbf{t} \tag{4.49}$$

where $\mathbb{W}$ *is a matrix with* $d_V \times d_S \times d_W$ *columns and* $d_V \times d_W$ *rows and the* n^{th} *row of* $\mathbb{W}$, $[\mathbb{W}]_n$, *has column entries*

$$[\mathbb{W}]_n^m = \begin{cases} 1, & m = \text{`}ijk\text{'} \\ 0, & m \neq \text{`}ijk\text{'} \end{cases} \tag{4.50}$$

where $\text{`}ijk\text{'}$ *denotes entry* $[[(i-1) \times d_S \times d_W] + k] + [(j-1) \times d_W]$ *for* $n = [(i-1) \times d_W] + k$ *and* $j = [1, d_S]$ *(i and k are fixed).*

Proof.

$$1_V \otimes \iota_S \otimes 1_W :: \mathbf{t} = \sum_{ijk} c_{ijk} \mathbf{v}_i \otimes \mathbf{s}_j \otimes \mathbf{w}_k \tag{4.51}$$

$$\mapsto \sum_{ijk} c_{ijk} \mathbf{v}_i \otimes \mathbf{w}_k \tag{4.52}$$

where

$$[\sum_{ijk} c_{ijk} \mathbf{v}_i \otimes \mathbf{w}_k]_n = \sum_j [\mathbf{t}]_{\text{`}ijk\text{'}} \tag{4.53}$$

$$= [\mathbb{W}]_n \cdot \mathbf{t} \tag{4.54}$$

(The result of applying the ι map is a tensor – or simple vector – where for each component we collapse, i.e. sum over, a given space.)

□

Proposition 4.3 (μ Map in **FVect** as Matrix-Vector Multiplication)**.** *For vectors*

$$\mathbf{v} = \sum_i c_i \mathbf{e}_i \tag{4.55}$$

$$\mathbf{w} = \sum_j c'_j \mathbf{e}_j \tag{4.56}$$

in a vector space V *with a fixed basis* $\{\mathbf{e}_i\}_i$, *the* μ *map is defined by*

$$\mu :: \mathbf{v} \otimes \mathbf{w} \mapsto \mathbb{W} \cdot \mathbf{w} \tag{4.57}$$

where

$$\mathbb{W} = \begin{bmatrix} c_1 & 0 & \cdots & 0 \\ 0 & c_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_{d_V} \end{bmatrix} \tag{4.58}$$

Proof.

$$\mu :: \mathbf{v} \otimes \mathbf{w} = (\sum_i c_i \mathbf{e}_i) \otimes (\sum_j c'_j \mathbf{e}_j) \tag{4.59}$$

$$= \sum_{ij} c_i \mathbf{e}_i \otimes c'_j \mathbf{e}_j \tag{4.60}$$

$$\mapsto \sum_{ij} \delta^j_i c_i c'_j \mathbf{e}_i \tag{4.61}$$

$$= \sum_i c_i c'_i \mathbf{e}_i \tag{4.62}$$

$$= \begin{bmatrix} c_1 & 0 & \cdots & 0 \\ 0 & c_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_{d_V} \end{bmatrix} \cdot \begin{bmatrix} c'_1 \\ c'_2 \\ \vdots \\ c'_{d_V} \end{bmatrix} \tag{4.63}$$

$$= \mathbb{W} \cdot \mathbf{w} \tag{4.64}$$

(When $\mathbf{v} \otimes \mathbf{w} \in V \otimes V$ is represented as a matrix, the result of applying the μ map is a vector in V consisting of only the diagonal elements of $\mathbf{v} \otimes \mathbf{w}$.)

□

Algorithm 2 Constructing ι in **FVect**

Require: $\mathbf{t}$ is a tensor (i.e. multi-dimensional array) in $V \otimes S \otimes W$

function $\iota(\mathbf{t})$
  $d_V \leftarrow \text{LENGTH}(\mathbf{t})$
  $d_S \leftarrow \text{LENGTH}(\mathbf{t}[1])$
  $d_W \leftarrow \text{LENGTH}(\mathbf{t}[1][1])$
  $n \leftarrow d_V \times d_W$ ▷ n: # of rows
  $m \leftarrow d_V \times d_S \times d_W$ ▷ m: # of columns
  $\mathbb{W} \leftarrow$ [[0 **for** 1 to m] **for** 1 to n]
  for $i \leftarrow 1$ to d_V **do**
    for $k \leftarrow 1$ to d_W **do**
      $r \leftarrow [(i-1) \times d_W] + k$ ▷ r: row
      $b \leftarrow [(i-1) \times d_S \times d_W] + k$ ▷ b: start column
      for $j \leftarrow 1$ to d_S **do**
        $c \leftarrow b + [(j-1) \times d_W]$ ▷ c: column
        $\mathbb{W}[r][c] \leftarrow 1$
  return $\mathbb{W}$ ▷ $\mathbb{W}$: *delete* matrix

The results presented in this subsection give us simple connectionist representations of information flow in compositional distributional semantics. These supply the basic operations at the heart of DISCOG. It is important to note that only

Algorithm 3 Constructing μ in **FVect**

Require: $\mathbf{v}$ is a vector of length n

```
function μ(v)
    W ← [[0 for 1 to n] for 1 to n]
    for i ← 1 to n do
        W[i][i] ← v[i]                    ▷ v[i]: c_i
    return W                              ▷ W: match matrix
```

matrices which encode entanglement, i.e. *reduce* matrices, are fit to be learned via backpropogation in the underlying connectionist network. (Our *delete* and *match* matrices solely serve to discard and pair information respectively and so are purely structural, following from the treatment of relative pronouns in [47].)

4.2.2 From Activation Values to Symbolic Structures

Now, we are ready to define the connectionist realization of a symbolic structure $\mathtt{s}$ within our adapted model of ICS based on the DISCO framework.

Definition 4.1 (Matrix-Vector Binding)**.** *The binding $\mathtt{f}/r$ of a filler $\mathtt{f}$ to a role r is realized as a vector $\mathbf{f}/\mathbb{W}_r$ that is the matrix-vector multiplication of a matrix $\mathbb{W}_r$ realizing r with a vector $\mathbf{f}$ realizing $\mathtt{f}$,*

$$\mathbf{f}/\mathbb{W}_r = \mathbb{W}_r \cdot \mathbf{f} \tag{4.65}$$

Definition 4.2 (Alternative Connectionist Realization)**.** *A symbolic structure $\mathtt{s}$ is defined by a collection of structural roles $\{r_i\}$ each of which may be occupied by a filler $\mathtt{f}_i$, where the realization $\mathbf{f}_1$ of $\mathtt{f}_1$ is the tensor product of a collection of realizations of atomic fillers $\{\mathbf{a}_j\}$ where $\mathbf{a}_j \in V_F$ (e.g. nouns in language). $\mathtt{s}$ is a set of constituents, each a filler/role binding $\mathtt{f}_i/r_i$. The connectionist realization of $\mathtt{s}$ is an activation vector*

$$\mathbf{s} = \mathbb{W}_{r_n} \cdot \mathbf{f}_n \tag{4.66}$$

that is the recursive matrix-vector multiplication of a matrix $\mathbb{W}_{r_i}$ realizing r_i with a vector $\mathbf{f}_i$ realizing a filler/role binding $\mathtt{f}_{i-1}/r_{i-1}$, i.e.

$$\begin{aligned} \mathbb{W}_{r_n} \cdot \mathbf{f}_n &= \mathbb{W}_{r_n} \cdot (\mathbb{W}_{n-1} \cdot \mathbf{f}_{r_{n-1}}) && (4.67)\\ &= \mathbb{W}_{r_n} \cdot (\mathbb{W}_{r_{n-1}} \cdot (\cdots \mathbb{W}_{r_1} \cdot (\mathbf{a}_1 \otimes \cdots \otimes \mathbf{a}_m) \cdots)) && (4.68) \end{aligned}$$

To unpack (4.68), $\mathbb{W}_{r_i}$ encode entanglement and realize pregroup reductions (i.e. information flow), while $\{\mathbf{a}_j\}$ provide *simple* distributional arguments (i.e. the building blocks of task-specific cognitive representations). As a sanity check, we can verify that our alternative connectionist realization holds for our example *Comedians tell funny jokes*. Our analysis of ϵ maps in **FVect** as matrix-vector multiplication tells us that

$$\begin{aligned}
\overrightarrow{\textit{Comedians tell funny jokes}} &= \mathbb{W}_{\text{tell}} \cdot (\mathbf{Comedians} \otimes \overrightarrow{\textit{funny jokes}}) && (4.69)\\
&= \mathbb{W}_{\text{tell}} \cdot (\mathbf{Comedians} \otimes [\mathbb{W}_{\text{funny}} \cdot \mathbf{jokes}]) && (4.70)\\
&= \mathbb{W}_{\text{tell}} \cdot ([\mathbb{I}_{d_N} \cdot \mathbf{Comedians}] \otimes [\mathbb{W}_{\text{funny}} \cdot \mathbf{jokes}]) && (4.71)\\
&= \mathbb{W}_{\text{tell}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{funny}}) \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) && (4.72)\\
&= \mathbb{W}_{\text{tell}} \cdot \mathbb{W}' \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) && (4.73)\\
&= \mathbb{W}'' \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) && (4.74)
\end{aligned}$$

where $\mathbb{W}' = \mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{funny}}$ and $\mathbb{W}'' = \mathbb{W}_{\text{tell}} \cdot \mathbb{W}'$. Note that (4.69) and (4.74) contrast the respective complex filler and complex role perspectives in our new model. Since the realizations of *Comedians tell jokes* and *Comedians tell funny jokes* now exist within a common semantic space thanks to the DISCO framework, we can measure their similarity using the cosine distance:

$$\frac{\langle \overrightarrow{\textit{Comedians tell jokes}} \mid \overrightarrow{\textit{Comedians tell funny jokes}} \rangle}{|\overrightarrow{\textit{Comedians tell jokes}}||\overrightarrow{\textit{Comedians tell funny jokes}}|} \tag{4.75}$$

Given our analysis from before, (4.75) reduces to:

$$\frac{\langle \overrightarrow{\textit{jokes}} \mid \overrightarrow{\textit{funny jokes}} \rangle}{|\overrightarrow{\textit{jokes}}||\overrightarrow{\textit{funny jokes}}|} \tag{4.76}$$

Hence, the similarity between the two sentences boils down to the similarity between *jokes* and *funny jokes*.

So what have we achieved with our DISCOG representation in relation to ICS? Well, the benefit of this new recursive representation is that filler/role bindings (i.e. constituents) living in different subspaces of $\mathcal{S}^*$ that make up a symbolic structure $\mathbf{s}$ are now composed in such a way that all well-formed $\mathbf{s}$, with respect to a certain cognitive task, are realized in a *finite shared* meaning space. This means that we have a model of representation in higher cognition that is a combination of filler/role bindings based on connectionist principles – as in ICS – which at the same time leverages the power of the DISCO framework to resolve the problem of

an unbounded representational space. Notably, the fillers and roles we consider in DISCOG are *genuine* meaning vectors (e.g. **Comedians**, **jokes**, $\mathbb{W}_{\text{tell}}$, etc.) as opposed to generic ones (e.g. **noun**, **verb**, $\mathbb{W}_{\mathtt{S}\to\mathtt{NP\ VP}}$, etc.) as in ICS. In this sense, DISCOG is a significant *extension* of ICS. In addition, our approach allows the comparison of well-formed symbolic structures with different underlying 'grammatical'[6] structures – a feature that ICS is unable to achieve. Ultimately, we can represent these matrices of compositional rules in a neural network and learn them via backpropagtion. This reflects (we believe) a practical development with respect to implementation of DISCO models. Looking at the bigger picture, the value in having general matrix representations of compositional rules for a particular cognitive task is that we can determine a formal equivalence between massively parallel distributed computations at a lower level in the cognitive architecture and arbitrarily complex (recursive) symbolic functions at a higher level in the cognitive architecture.

4.2.3 Toy Examples

Here, we walk through a number of toy linguistic examples to give the reader a chance to verify the results presented in the previous subsections and to clarify any confusion over the definitions of binding and connectionist realization in DISCOG. We use simple toy distributional representations shown in (4.77) for our computations. We assume $N = \mathbb{R}^2 = S$ and let $\{\mathbf{n}_i\}_i$ and $\{\mathbf{s}_j\}_j$ denote orthonormal bases of N and S respectively. **comedians**, **jokes** $\in N$, **tell** $\in N \otimes S \otimes N$, and **funny** $\in N \otimes N$. Reading (4.77) is straightforward, e.g. $[\mathbf{jokes}]_1 = 5$, $[\mathbf{tell}]_{212} = 9$, $[\mathbf{funny}]_{12} = 7$, and so on.

comedians	*jokes*
7	5
4	1

tell			
3	8	4	1
6	2	9	5

funny	
2	6
7	3

(4.77)

Example 4.1 (Simple Positive Transitive Sentence)**.** Consider the sentence

$$\textit{Comedians tell jokes.} \tag{4.78}$$

From (3.48), we know that the meaning vector of (4.78) is:

$$\overrightarrow{\textit{Comedians tell jokes}} = \sum_k \mathbf{s}_k \sum_{il} c_i^{\mathbf{Comedians}} c_{ikl}^{\mathbf{tell}} c_l^{\mathbf{jokes}} \tag{4.79}$$

[6]Here, 'grammatical' means the rules via which fillers may be composed in any cognitive task that can be decomposed into a set of filler/role bindings, not limited to language understanding.

$$= \begin{bmatrix} 7 \cdot [(3 \cdot 5) + (6 \cdot 1)] + 4 \cdot [(4 \cdot 5) + (9 \cdot 1)] \\ 7 \cdot [(8 \cdot 5) + (2 \cdot 1)] + 4 \cdot [(1 \cdot 5) + (5 \cdot 1)] \end{bmatrix} \tag{4.80}$$

$$= \begin{bmatrix} 263 \\ 334 \end{bmatrix} \tag{4.81}$$

From (4.66), we know that the meaning vector of (4.78) is:

$$\overrightarrow{\textit{Comedians tell jokes}} = \mathbb{W}_{\text{tell}} \cdot (\mathbf{Comedians} \otimes \mathbf{jokes}) \tag{4.82}$$

$$= \epsilon(\mathbf{tell}) \cdot \begin{bmatrix} 7 \cdot 5 \\ 7 \cdot 1 \\ 4 \cdot 5 \\ 4 \cdot 1 \end{bmatrix} \tag{4.83}$$

$$= \begin{bmatrix} 3 & 6 & 4 & 9 \\ 8 & 2 & 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} 35 \\ 7 \\ 20 \\ 4 \end{bmatrix} \tag{4.84}$$

$$= \begin{bmatrix} 263 \\ 334 \end{bmatrix} \tag{4.85}$$

Hence, (4.81) and (4.86) are the same.

Example 4.2 (Complex Positive Transitive Sentence)**.** Consider the sentence

$$\textit{Comedians tell funny jokes.} \tag{4.86}$$

From (3.57), we know that the meaning vector of (4.87) is:

$$\overrightarrow{\textit{Comedians tell funny jokes}} = \sum_k \mathbf{s}_k \sum_{iln} c_i^{\mathbf{Comedians}} c_{ikl}^{\mathbf{tell}} c_{ln}^{\mathbf{funny}} c_n^{\mathbf{jokes}} \tag{4.87}$$

$$= \sum_k \mathbf{s}_k \sum_{il} c_i^{\mathbf{Comedians}} c_{ikl}^{\mathbf{tell}} c_l^{\overrightarrow{\textit{funny jokes}}} \tag{4.88}$$

$$= \begin{bmatrix} 7 \cdot [(3 \cdot 17) + (6 \cdot 33)] + 4 \cdot [(4 \cdot 17) + (9 \cdot 33)] \\ 7 \cdot [(8 \cdot 17) + (2 \cdot 33)] + 4 \cdot [(1 \cdot 17) + (5 \cdot 33)] \end{bmatrix} \tag{4.89}$$

$$= \begin{bmatrix} 3203 \\ 2142 \end{bmatrix} \tag{4.90}$$

where

$$\overrightarrow{\textit{funny jokes}} = \sum_i \mathbf{n}_i \sum_j c_{ij}^{\mathbf{funny}} c_j^{\mathbf{jokes}} \tag{4.91}$$

$$= \begin{bmatrix} (2 \cdot 5) + (7 \cdot 1) \\ (6 \cdot 5) + (3 \cdot 1) \end{bmatrix} \tag{4.92}$$

$$= \begin{bmatrix} 17 \\ 33 \end{bmatrix} \tag{4.93}$$

From (4.66), we know that the meaning vector of (4.87) is:

$$\overrightarrow{\textit{Comedians tell funny jokes}} = \mathbb{W}_{\text{tell}} \cdot ((\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{funny}}) \cdot (\mathbf{Comedians} \otimes \mathbf{jokes})) \tag{4.94}$$

$$= \epsilon(\mathbf{tell}) \cdot (\mathbb{I}_{d_N} \otimes \epsilon(\mathbf{funny})) \cdot (\mathbf{Comedians} \cdot \mathbf{jokes}) \tag{4.95}$$

$$= \begin{bmatrix} 3 & 6 & 4 & 9 \\ 8 & 2 & 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} 2 & 7 & 0 & 0 \\ 6 & 3 & 0 & 0 \\ 0 & 0 & 2 & 7 \\ 0 & 0 & 6 & 3 \end{bmatrix} \cdot \begin{bmatrix} 35 \\ 7 \\ 20 \\ 4 \end{bmatrix} \tag{4.96}$$

$$= \begin{bmatrix} 42 & 39 & 44 & 55 \\ 28 & 62 & 32 & 22 \end{bmatrix} \cdot \begin{bmatrix} 35 \\ 7 \\ 20 \\ 4 \end{bmatrix} \tag{4.97}$$

$$= \begin{bmatrix} 3203 \\ 2142 \end{bmatrix} \tag{4.98}$$

Hence, (4.90) and (4.98) are the same.

Example 4.3 (Subject Relative Pronoun)**.** Consider the phrase

$$\textit{Comedians who tell jokes.} \tag{4.99}$$

From (3.81), we know that the meaning vector of (4.99) is:

$$\overrightarrow{\textit{Comedians who tell jokes}} = \sum_i c_i^{\mathbf{Comedians}} \mathbf{n}_i \sum_{kl} c_{ikl}^{\mathbf{tell}} c_l^{\mathbf{jokes}} \tag{4.100}$$

$$= \begin{bmatrix} 7 \cdot [(3 \cdot 5) + (6 \cdot 1) + (8 \cdot 5) + (2 \cdot 1)] \\ 4 \cdot [(4 \cdot 5) + (9 \cdot 1) + (1 \cdot 5) + (5 \cdot 1)] \end{bmatrix} \tag{4.101}$$

$$= \begin{bmatrix} 441 \\ 156 \end{bmatrix} \tag{4.102}$$

Let $\mathbf{tell}^{Obj}$ indicate that the role **tell** composes only with an *object* argument. As such, our ϵ function assigns $n \leftarrow d_V \times d_S$ and $m \leftarrow 1$ on lines 5 and 6 of the algorithm respectively – think of the argument tensor as $\mathbf{r} \in 1 \otimes (V \otimes S) \otimes W$. From (4.66), we know that the meaning vector of (4.99) is:

$$\overrightarrow{\textit{Comedians who tell jokes}} = \mathbb{W}_{\text{Comedians}} \cdot (\mathbb{W}_{\iota} \cdot (\mathbb{W}_{\text{tell}}^{Obj} \cdot (\mathbf{jokes}))) \tag{4.103}$$

$$= \mu(\mathbf{Comedians}) \cdot \iota(\mathbf{tell}^{Obj}) \cdot \epsilon(\mathbf{tell}^{Obj}) \cdot \begin{bmatrix} 5 \\ 1 \end{bmatrix} \tag{4.104}$$

$$= \begin{bmatrix} 7 & 0 \\ 0 & 4 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 & 6 \\ 8 & 2 \\ 4 & 9 \\ 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 1 \end{bmatrix} \tag{4.105}$$

$$= \begin{bmatrix} 7 & 7 & 0 & 0 \\ 0 & 0 & 4 & 4 \end{bmatrix} \cdot \begin{bmatrix} 3 & 6 \\ 8 & 2 \\ 4 & 9 \\ 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 1 \end{bmatrix} \tag{4.106}$$

$$= \begin{bmatrix} 77 & 56 \\ 20 & 56 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 1 \end{bmatrix} \tag{4.107}$$

$$= \begin{bmatrix} 441 \\ 156 \end{bmatrix} \tag{4.108}$$

Hence, (4.108) and (4.102) are the same.

Example 4.4 (Object Relative Pronoun)**.** Consider the phrase

$$\textit{Jokes which comedians tell.} \tag{4.109}$$

From (3.89), we know that the meaning vector of (4.109) is:

$$\overrightarrow{\textit{Jokes which comedians tell}} = \sum_i c_i^{\mathbf{Jokes}} \mathbf{n}_i \sum_{jl} c_j^{\mathbf{comedians}} c_{jli}^{\mathbf{tell}} \tag{4.110}$$

$$= \begin{bmatrix} 5 \cdot [(7 \cdot 3) + (7 \cdot 8) + (4 \cdot 4) + (4 \cdot 1)] \\ 1 \cdot [(7 \cdot 6) + (7 \cdot 2) + (4 \cdot 9) + (4 \cdot 5)] \end{bmatrix} \tag{4.111}$$

$$= \begin{bmatrix} 485 \\ 112 \end{bmatrix} \tag{4.112}$$

Let $\mathbf{tell}^{Subj}$ indicate that the role **tell** composes only with a *subject argument*. As such, our ϵ function assigns $n \leftarrow d_W \times d_S$ and $m \leftarrow 1$ on lines 5 and 6 of the algorithm respectively – think of the argument tensor as $\mathbf{r} \in 1 \otimes (W \otimes S) \otimes V$, hence as a preprocessing step we switch subject and object indices (i.e. $[\mathbf{tell}^{Subj}]_{kji} = [\mathbf{tell}]_{ijk}$). From (4.66), we know that the meaning vector of (4.109) is:

$$\overrightarrow{\textit{Jokes which comedians tell}} = \mathbb{W}_{\text{Jokes}} \cdot (W_\iota \cdot (W_{\text{tell}}^{Subj} \cdot (\mathbf{comedians}))) \tag{4.113}$$

$$= \mu(\mathbf{Jokes}) \cdot \iota(\mathbf{tell}^{Subj}) \cdot \epsilon(\mathbf{tell}^{Subj}) \cdot \begin{bmatrix} 7 \\ 4 \end{bmatrix} \tag{4.114}$$

$$= \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 & 4 \\ 8 & 1 \\ 6 & 9 \\ 2 & 5 \end{bmatrix} \cdot \begin{bmatrix} 7 \\ 4 \end{bmatrix} \tag{4.115}$$

$$= \begin{bmatrix} 5 & 5 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 & 4 \\ 8 & 1 \\ 6 & 9 \\ 2 & 5 \end{bmatrix} \cdot \begin{bmatrix} 7 \\ 4 \end{bmatrix} \tag{4.116}$$

$$= \begin{bmatrix} 55 & 25 \\ 8 & 14 \end{bmatrix} \cdot \begin{bmatrix} 7 \\ 4 \end{bmatrix} \tag{4.117}$$

$$= \begin{bmatrix} 485 \\ 112 \end{bmatrix} \tag{4.118}$$

Hence, (4.118) and (4.112) are the same.

As such, these examples should give the reader confidence in our passage from the DisCo model of semantics to DisCog. It should be evident that our set of elementary connectionist operations allows for an effortless application of the compositional distributional approach to ICS on account of our filler/role perspective. Significantly, this approach enables the mapping of realizations of same-type symbolic structures made up of filler/role bindings to a shared meaning space independent of 'grammatical' structure, e.g. (4.78) and (4.86). What's more, DisCog exploits the reductions (i.e. collapsing of high-dimensional vector spaces) of the DisCo model to solve the headache of an unbounded representational space $\mathcal{S}^*$ in ICS, while at the same time extending filler representation to authentic meaning vectors.

4.3 Unbinding Problem

One of the central assumptions of ICS is the respective linear independence of realizations of filler and roles. This assumption guarantees that fillers and roles form bases of their respective spaces which allows for a simple unbinding procedure. The availability of an unbinding mechanism is essential to the hypothesis that connecetionist algorithms can precisely realize symbolic functions at a higher-level in the cognitive architecture. Here, we detail the problems of the linear independence assumption in DisCog and offer a provisional solution.

4.3.1 A Case against Linear Independence

For a vector space V of dimension n, there can be at most n linearly independent vectors. When we think about distributional representations this becomes a major problem because the number of representations in a given space most certainly exceeds the size of the space. As an example, the evolution of language which

generates an ever-expanding lexicon highlights the falsehood of such an assumption in the context of distributional word representation. The reason why [52] puts forward the linear independence assumption is because the fillers considered in ICS are of generic form, e.g. noun, verb, sentence, and so on. In this setting, there are exponentially fewer fillers, so it is reasonable to claim that they are linearly independent. However, in that DISCOG is an extension of ICS working with bona fide meaning vectors, we cannot enforce such a requirement without greatly restricting say our vocabulary within a linguistic task. Ultimately, the issue of linear independence is a trade-off between space and expressiveness. Since our model exists in a finite-dimensional vector space corresponding to a finite connectionist network, we will need some notion of approximate unbinding to recover many of the important features of ICS.

4.3.2 Approximate Unbinding

In DISCOG, symbolic representations are realized via filler/role bindings. A role r is realized as a matrix $\mathbb{W}_r$ and a filler $\mathbf{f}$ is realized as a vector $\mathbf{f}$. Unbinding is the procedure where we extract a filler from a binding. To achieve this, we require a method to invert $\mathbb{W}_r$. Inverting $\mathbb{W}_r$ poses some difficulty because we cannot ensure $\mathbb{W}_r$ is invertible! The majority of $\mathbb{W}_r$ aren't even square, e.g. $\mathbb{W}_{\text{tell}}$ which has $dim(N) \times dim(N)$ columns and $dim(S)$ rows (assuming $dim(N) \times dim(N) \neq dim(S)$). Given this, we propose using the generalized inverse for performing approximate unbinding [35].

Definition 4.3 (Moore-Penrose Pseudo Inverse)**.** *For* $\mathbb{M} \in \mathbb{R}^{m \times n}$*, there exists a unique* $\mathbb{M}^+ \in \mathbb{R}^{n \times m}$ *such that the following conditions are satisfied:*

1. $\mathbb{M} \cdot \mathbb{M}^+ \cdot \mathbb{M} = \mathbb{M}$
2. $\mathbb{M}^+ \cdot \mathbb{M} \cdot \mathbb{M}^+ = \mathbb{M}^+$
3. $(\mathbb{M} \cdot \mathbb{M}^+)^{\mathrm{T}} = \mathbb{M} \cdot \mathbb{M}^+$
4. $(\mathbb{M}^+ \cdot \mathbb{M})^{\mathrm{T}} = \mathbb{M}^+ \cdot \mathbb{M}$

While the proofs of these equations is beyond the scope of this work, we note some pertinent properties of the Moore-Penrose pseudo inverse:

1. If $m = n$ and $\mathbb{M}$ is full rank, $\mathbb{M}^+ = \mathbb{M}^{-1}$.

2. For

$$\mathbb{M} \cdot \mathbf{a} = \mathbf{b} \tag{4.119}$$

where $\mathbb{M} \in \mathbb{R}^{m \times n}$, $\mathbf{a} \in \mathbb{R}^n$, and $\mathbf{b} \in \mathbb{R}^m$

(a) If $m > n$, $\mathbf{c} = \mathbb{M}^+ \cdot \mathbf{b}$ is the solution that minimizes $||\mathbf{b} - (\mathbb{M} \cdot \mathbf{c})||$. In other words, the Moore-Penrose pseudo inverse provides the least squares solution to (4.119), i.e. $\mathbf{c}$ is such that $\mathbb{M} \cdot \mathbf{c}$ is the solution 'closest' to the desired solution vector $\mathbf{b}$.

(b) If $m < n$, $\mathbf{c} = \mathbb{M}^+ \cdot \mathbf{b}$ is the solution that minimizes $||\mathbf{c}||$.

Using the Moore-Penrose pseudo inverse, we can now define our approximate unbinding procedure.

Definition 4.4 (Approximate Unbinding)**.** *For a binding* $\mathtt{f}/r$ *of a filler* $\mathtt{f}$ *to a role* r *realized as a vector* $\mathbf{f}/\mathbb{W}_r$ *that is the matrix-vector multiplication of a matrix* $\mathbb{W}_r$ *realizing* r *with a vector* $\mathbf{f}$ *realizing* $\mathtt{f}$*, i.e.* $\mathbb{W}_r \cdot \mathbf{f}$*, we approximately unbind* $\mathtt{f}$ *from* r *by application of the Moore-Penrose pseudo inverse of* $\mathbb{W}_r$

$$\mathbb{W}_r^+ \cdot (\mathbb{W}_r \cdot \mathbf{f}) \approx \mathbf{f} \tag{4.120}$$

4.4 Parallel Distributed Processing

Given our presentation of an alternative connectionist realization and the availability of an approximate unbinding procedure, it now seems appropriate to illustrate how connectionist principles in our new model percolate up to the symbolic level. The importance of this property is that we ensure our model can compute arbitrarily complex (recursive) symbolic functions in a massively parrallel distributed fashion just like in ICS. As an example, we will define a simple function $\mathtt{f}$ that takes a symbolic structure $\mathtt{s}$ representing a simple transitive sentence and returns a symbolic structure $\mathtt{t}$ representing a complex transitive sentence where an adjective is applied to the object of $\mathtt{s}$.

Example 4.5 (Object Adjective Application)**.** Let the realization $\mathbf{s}$ of a simple positive transitive sentence $\mathtt{s}$ be

$$\mathbf{s} = \mathbb{W}_{\text{trans}} \cdot (\mathbf{subj} \otimes \mathbf{obj}) \tag{4.121}$$

where $\mathbf{subj}, \mathbf{obj} \in N$. To construct a complex positive transitive sentence $\mathtt{t}$, we apply an adjective realized by a matrix $\mathbb{W}_{\text{adj}}$ to $\mathbf{obj}$:

$$\mathbf{t} = \mathbb{W}_{\text{trans}} \cdot (\mathbf{subj} \otimes [\mathbb{W}_{\text{adj}} \cdot \mathbf{obj}]) \tag{4.122}$$

Claim 4.1 (Connectionist Realization of a Symbolic Function)**.** *The connectionist realization of the symbolic function* $\mathtt{f} :: \mathtt{s} \to \mathtt{t}$ *is defined by*

$$\mathcal{F}(\mathbb{W}_{\text{adj}}, \mathbf{s}) = \mathbb{W}_{\mathcal{F}} \cdot \mathbf{s} \tag{4.123}$$

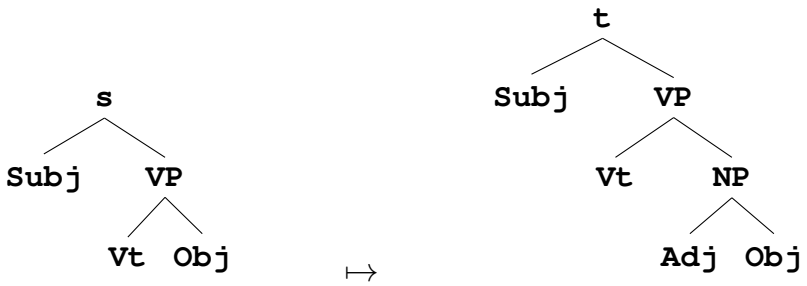
where

$$\mathbb{W}_{\mathcal{F}} = \mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot \mathbb{W}^{+}_{\text{trans}} \tag{4.124}$$

Proof.

$$\begin{aligned}
\mathbb{W}_{\mathcal{F}} \cdot \mathbf{s} &= [\mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot \mathbb{W}^{+}_{\text{trans}}] \cdot \mathbf{s} && (4.125)\\
&= [\mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot \mathbb{W}^{+}_{\text{trans}}] \cdot [\mathbb{W}_{\text{trans}} \cdot (\mathbf{subj} \otimes \mathbf{obj})] && (4.126)\\
&= \mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot (\mathbb{W}^{+}_{\text{trans}} \cdot \mathbb{W}_{\text{trans}}) \cdot (\mathbf{subj} \otimes \mathbf{obj}) && (4.127)\\
&\approx \mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot \mathbb{I}_{d_N \times d_N} \cdot (\mathbf{subj} \otimes \mathbf{obj}) && (4.128)\\
&= \mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \otimes \mathbb{W}_{\text{adj}}) \cdot (\mathbf{subj} \otimes \mathbf{obj}) && (4.129)\\
&= \mathbb{W}_{\text{trans}} \cdot (\mathbb{I}_{d_N} \cdot \mathbf{subj}) \otimes (\mathbb{W}_{\text{adj}} \cdot \mathbf{obj}) && (4.130)\\
&= \mathbb{W}_{\text{trans}} \cdot (\mathbf{subj} \otimes [\mathbb{W}_{\text{adj}} \cdot \mathbf{obj}]) && (4.131)\\
&= \mathbf{t} && (4.132)
\end{aligned}$$

Graphically, we can represent the symbolic function $\mathtt{f}$ as (4.133):



(4.133)

Comedians tell jokes *Comedians tell funny jokes*

□

Hence, we can see how massively distributed and parallel spreading activation in our underlying connectionist network realizes symbolic step-by-step algorithms at the higher-level substrate of our cognitive architecture.

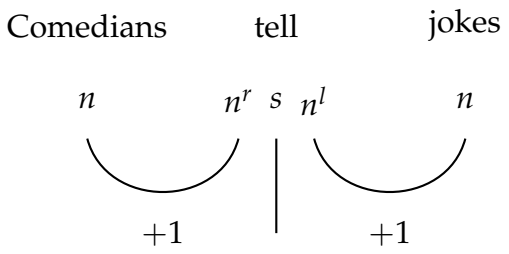
4.5 Functional Harmony

One of the qualities of ICS is the characterization of the connectionist notion of well-formedness, i.e. *harmony*. Harmony is a measure of the degree to which an activation vector – or flow of information – in a connectionist network respects the constraints encoded in the connections. To return back to our linguistic instance of a cognitive task, we can think of the harmony of the realization $\mathbf{s}$ of a string of words $\mathbf{s}$ as the extent to which $\mathbf{s}$ is grammatical, where the connectionist network in this case is built from the fundamental matrices encoding grammatical compositional rules which have been learned prior. In this sense, lower-level connectionist principles inform analysis at a higher level in the cognitive architecture.

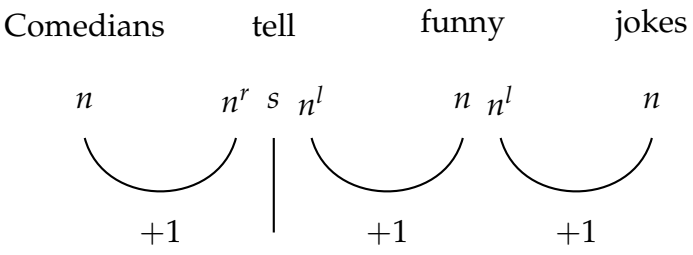
Conceptually, the matrices $\mathbb{W}$ in ICS that encode a set of harmonic constraints pick out constituents in a representation $\mathbf{s}$ that satisfy the *general* compositional rules of a 'grammar' – rules similar to those found in formal semantics. As we have seen, the harmony of an activation vector is in fact equivalent to the harmony of co-occurring constituents $(\mathbf{c}_i, \mathbf{c}_j)$ in a symbolic structure $\mathbf{s}$. In our framework, a string of words $\mathbf{s}$ is grammatical if and only if it reduces to type s (i.e. a sentence), where reductions are defined by a pregroup grammar. As such, it suffices to define harmony from a higher level of analysis with respect to pregroup grammar and work our way down to a connectionist definition.

4.5.1 Harmonic Symbols

In the DISCO model of semantics, grammatical rules are implemented via linear maps. Therefore, it seems natural that our notion of harmony should be shaped in some manner by harmony values that are assigned to each of the available linear maps. A naive harmonic grammar could assign one point to each ϵ reduction:

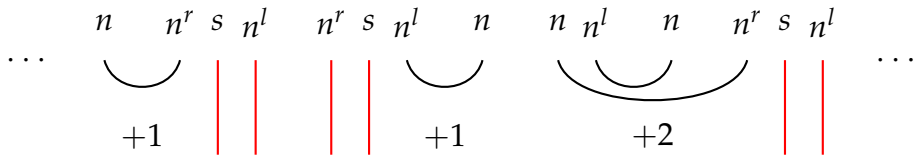


(4.134)



(4.135)

Given this definition, *Comedians tell funny jokes* (+3) would be more grammatical than *Comedians tell jokes* (+2). From a purely symbolic perspective, it doesn't seem intuitive to claim that (4.135) is more well-formed than (4.134). Ultimately, both strings of words reduce to a sentence and therefore should be deemed equally grammatical. That being said, from a connectionist – or even aesthetic – perspective, there is an argument that a complex well-formed structure is more harmonic than a simple well-formed structure since it satisfies more constraints. As an analogy, consider two computer programs. The first is an empty `main()` function. The second is an implementation of VIRTUALMEMORY. Both programs compile. Symbolically, a compiler makes no difference between its grammatical assessments of the programs. However, at runtime, it would seem that the *realizations* of the programs at a lower level would warrant a more sophisticated notion of harmony that accounts for complexity. But just as well, it is common thought that the simplest programs are the most beautiful[7]! In this respect, we are hard-pressed to come up with a faithful quantification of harmony because, simply put, it is hard to describe. Nevertheless, returning to (4.134) and (4.135), it would appear that our naive harmonic grammar is inadequate because ungrammatical sentences with a large number of ϵ reductions are assigned high harmony:

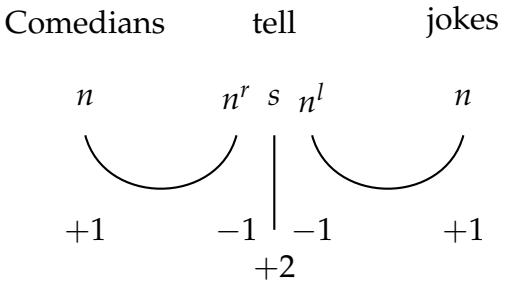


(4.136)

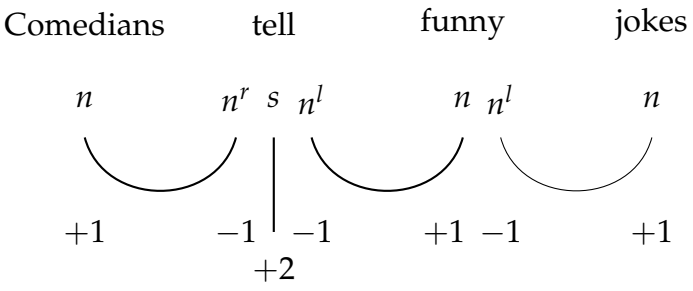
An obvious solution to this problem is to reframe the goal of harmony maximization in ICS to the objective of *harmony balancing* in DISCOG. In this formulation, a symbolic structure is considered well-formed if its harmony value is close to some point of equilibrium. As such, we could assign positive values to atomic types (e.g. n, s, etc.) and corresponding negative values to adjoints (e.g. n^l, s^r, etc.). Specifically, we want the harmony of a well-formed symbol **s** to evaluate to the

[7]We make the assumption that the word *beautiful* in some sense captures a notion of well-formedness.

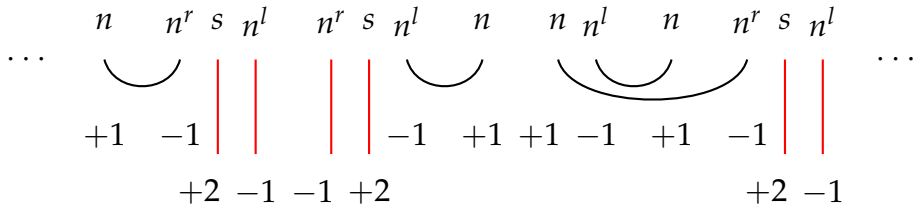
harmony allocated to the type considered grammatical in a certain cognitive task. So for language, all well-formed strings of words should have harmony equal to the harmony value assigned to the type s (i.e. a sentence). We can assign different values to different types to help us distinguish symbolic structures of one type from those of another:



(4.137)



(4.138)

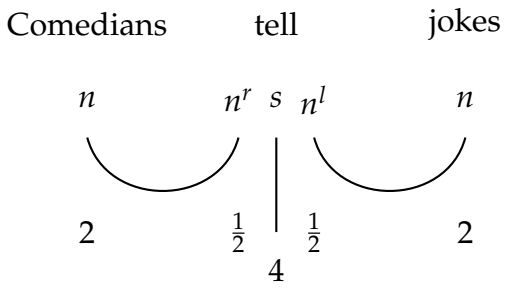


(4.139)

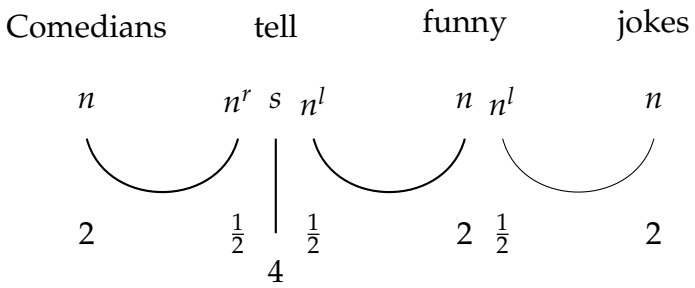
This kind of scheme attempts to identify a bijective correspondence between fillers and roles in a symbolic structure as the condition for optimal harmony. As motivation for this definition, let us briefly remark on our interpretation of fillers as arguments to functions pointed to by roles. A function is properly invoked when the correct parameters are supplied. Since roles act as an API interfacing the lower- and higher-level layers of our cognitive architecture, for a binding to be properly realized this computational contract must be respected. Therefore, it seems reasonable to give an account of harmony that rests on the accurate pairing of arguments with argument slots, i.e. fillers with roles. However, as we can see in (4.139), a simple summation of harmony values does not capture the notion of *correct pairing* due to the commutative property of addition. A collection of harmony values

$\{+1, +1, +1\}$ – as in (4.139) – indicates the presence of three unbound nouns as just one possible set of symbolic structures within an infinite space of variations. Note that these structures do not *interact*, hence in the *functional* setting of harmony balancing, we consider these harmonies *separate* (more on this coming up).

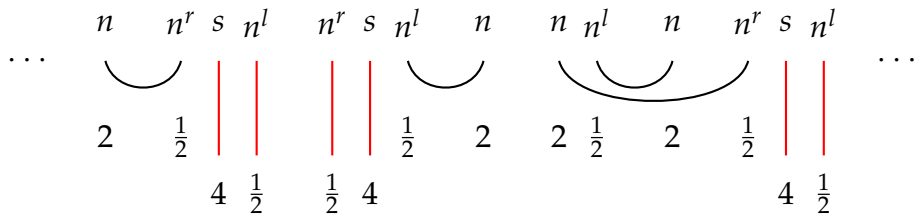
Nonetheless, it is important to remind ourselves that although well-formedness is a strict notion, harmony is simply a *graded measure* that should inform a network how to fix an activation vector. Given this, we recommend proceeding with our notion of harmony balancing which seems to be a more fitting characterization of well-formedness in comparison to harmony maximization. One thing we must alter though is the operation by which harmony values are combined. Since connectionist realization in DISCOG is recursive on matrix-vector binding, the operation used to compose harmony values should be multiplication. We model harmony balancing by mapping positive harmony values to values greater than 1 and negative harmony values to corresponding (reciprocal) values between 0 and 1. The net effect is the same as in the additive case: the harmony of a well-formed symbolic structure is equal to the harmony value assigned to the first-class type of a cognitive task (e.g. $H(s)$ in language). We use harmony values which are powers of 2 for ease of representation and efficiency of bitwise operations (with a view to implementation at the connectionist level):



(4.140)



(4.141)



(4.142)

Using this multiplicative notion of harmony balancing – which we refer to as *functional harmony* – we now give a working definition of harmony in DISCOG.

Definition 4.5 (Functional Harmony of a Symbolic Representation)**.** *Suppose* $\mathbf{s}$ *is a symbolic structure with constituents* $\{\mathbf{c}_i\}$*. Then the functional harmony of* $\mathbf{s}$ *is defined by*

$$H(\mathbf{s}) = \times_i H(\mathbf{c}_i) \tag{4.143}$$

where $H(\mathbf{c}_i)$ *– the harmony resulting from the occurrence of* $\mathbf{c}_i$ *– is a constant for all* $\mathbf{s}$*.*

Although not obvious, our definition of harmony fundamentally rests on compositional principles. The harmony of a constituent, i.e. $H(\mathbf{c}_i)$, is defined in *relation* to other constituents $\mathbf{c}_j$ via our recursive definition of connectionist realization. Suppose the harmony of a (grammatical) transitive sentence $\mathbf{s}$ is 4. Then the harmony of the fillers it is composed of must multiply to 4, i.e. $H(\mathbf{s}) = H(\mathbf{f}_1) \times H(\mathbf{f}_2)$. This is because the harmony of a binding represents how well its constituents compose, and the method of composition in DISCOG is matrix-vector binding. As such, we can think of roles in well-formed structures as the multiplication operators ($\times$) in (4.143). To elaborate, roles do not 'have' harmony in as much as they serve to inform the relationship by which we can assess the well-formedness of filler composition. Examining (4.141), we see that the harmony values 'assigned' to *tell* and *funny* evaluate to 1. 1 is the unit of multiplication. Therefore, roles have no effect on the total harmony of a symbolic structure beyond advising what harmonies of argument fillers are required to achieve the most harmonic complex filler upon binding, i.e. a state of equilibrium.

One might worry that our definition allows for a trivial notion of harmony since if the harmony of a symbolic structure is just the multiplication of constituent harmonies, any set of the right number of atomic fillers results in a 'well-formed' structure. However, this is not the case. We remind the reader that symbolic structures are formed via a *binding* mechanism. Thus, if there is no binding, there is no recursion, and there can only be one constituent in the structure. (To say otherwise would be equivalent to claiming that an atomic symbol is complex!) It is in this sense that our definition of harmony is functional.

Here, we give some simple harmony calculations for further clarification. We assume that a phrase is well-formed (i.e. a sentence) if it is *composed* of two nouns. We denote our harmonic grammar by

$$H_{\mathcal{G}} \equiv \{H(s) = H(n) \times H(n)\} \tag{4.144}$$

The absence of a multiplication operation ($\times$) between harmony values indicates harmonies of *separate* symbolic structures – so we can think of our harmony function as returning a set of harmony values, i.e. one for each structure.

Example 4.6 (Complex Positive Transitive Sentence)**.** Consider the phrase

Comedians tell funny jokes. (4.145)

Let $\mathbf{s}$ represent (4.145). The harmony of $\mathbf{s}$ is defined by

$$\begin{aligned}
H(\mathbf{s}) &= \times_i H(\mathbf{c}_i) && (4.146)\\
&= H(Comedians) \times H(tell) \times H(funny\ jokes) && (4.147)\\
&= H(Comedians) \times 1 \times [H(funny) \times H(jokes)] && (4.148)\\
&= H(Comedians) \times [1 \times H(jokes)] && (4.149)\\
&= H(Comedians) \times H(jokes) && (4.150)\\
&= H(n) \times H(n) && (4.151)
\end{aligned}$$

Hence, (4.145) is well-formed.

Example 4.7 (Noun Phrase)**.** Consider the phrase

Funny jokes. (4.152)

Let $\mathbf{s}$ represent (4.152). The harmony of $\mathbf{s}$ is defined by

$$\begin{aligned}
H(\mathbf{s}) &= \times_i H(\mathbf{c}_i) && (4.153)\\
&= H(Funny\ jokes) && (4.154)\\
&= H(Funny) \times H(jokes) && (4.155)\\
&= 1 \times H(jokes) && (4.156)\\
&= H(jokes) && (4.157)\\
&= H(n) && (4.158)
\end{aligned}$$

Hence, (4.152) is not well-formed. $H(\mathbf{s})$ informs us that to mend $\mathbf{s}$ we *probably* need some role that allows $\mathbf{s}$ to compose with another noun. (This wouldn't be the case if $\mathbf{s}$ represented *Comedians tell* because we would just need to *insert* a right-sided noun phrase into $\mathbf{s}$.)

Example 4.8 (Juxtaposed Nouns)**.** Consider the phrase

$$\textit{Jokes comedians.} \tag{4.159}$$

Let $\mathbf{s}$ represent (4.159). The harmony of $\mathbf{s}$ is defined by

$$\begin{aligned} H(\mathbf{s}) &= \times_i H(\mathbf{c}_i) && (4.160)\\ &= H(Jokes) \quad H(comedians) && (4.161)\\ &= \{H(n), H(n)\} && (4.162)\\ &\equiv \{H(\mathbf{s}_1), H(\mathbf{s}_2)\} && (4.163) \end{aligned}$$

Hence, (4.159) is not well-formed. $H(\mathbf{s})$ informs us that to mend $\mathbf{s}$ we *probably* need some role that allows $\mathbf{s}_1$ to compose with $\mathbf{s}_2$. (This wouldn't be the case if $\mathbf{s}_2$ represented *Comedians tell* because we would just need to *permute* the order of structures in $\mathbf{s}$.)

Example 4.9 (Misplaced Adjective)**.** Consider the phrase

$$\textit{Comedians funny tell jokes.} \tag{4.164}$$

Let $\mathbf{s}$ represent (4.164). The harmony of $\mathbf{s}$ is defined by

$$\begin{aligned} H(\mathbf{s}) &= \times_i H(\mathbf{c}_i) && (4.165)\\ &= H(Comedians) \quad H(funny) \quad [H(tell) \times H(jokes)] && (4.166)\\ &= H(n) \quad 1 \quad [1 \times H(n)] && (4.167)\\ &= H(n) \quad 1 \quad H(n) && (4.168)\\ &= \{H(n), 1, H(n)\} && (4.169)\\ &\equiv \{H(\mathbf{s}_1), H(\mathbf{s}_2), H(\mathbf{s}_3)\} && (4.170) \end{aligned}$$

Hence, (4.164) is not well-formed. $H(\mathbf{s})$ informs us that to mend $\mathbf{s}$ we *probably* need some role other than $\mathbf{s}_2$ that allows $\mathbf{s}_1$ to compose with $\mathbf{s}_3$. (In fact, we just need to *delete* $\mathbf{s}_2$ from $\mathbf{s}$.)

4.5.2 Harmonic Activation Values

Given the definition of harmony in ICS, the passage to a connectionist definition of harmony in DISCOG would seem rather difficult. Let us recall that a symbolic structure $\mathbf{s}$ is realized as an activation vector $\mathbf{s}$ via a filler/role binding $\mathbf{f}/\mathbb{W}_r$. As such, extracting constituents – which is required for harmony evaluation in

ICS – amounts to the task of unbinding fillers from roles at different levels in a symbolic structure (i.e. subspaces of $\mathcal{S}^*$). As we have previously discussed, unbinding is problematic for DISCOG because unlike ICS our model of cognition extends to genuine meaning vectors which makes it unfeasible to assume linear independence. The rule matrices considered in [52] are generic in that they represent compositional rules for *all* nouns, verbs, and so on. This entails that the meaning space under consideration is very small indeed. Because of this, specific rules can be written out for each construction in a grammar and easily applied. But with meaning vectors – as in our case – we can't write out a different rule for each possible combination of meanings. For example, to *replicate* a compositional rule decomposing generic sentences into generic nouns and generic verbs in ICS (i.e. $\mathbb{W}_{\texttt{S}\rightarrow\texttt{NP VP}}$), we would have to put together a list of rules for every possible combination of nouns and verbs that results in a sentence in DISCOG (i.e. $\{\mathbb{W}_{\text{tell}\rightarrow\text{Comedians jokes}}, \mathbb{W}_{\text{tell}\rightarrow\text{Reporters facts}}, \cdots\}$)! So, not only does our extension to meaning vectors present problems in terms of precise unbinding, but also in terms of formulating harmonic grammars in an ICS-like manner at the connectionist level.

That being said, if a harmonic grammar were to be implemented in DISCOG at the connectionist level, we would simply apply the weights of the harmony values at the symbolic level to corresponding role matrices at the lower level, e.g. $w_{\text{tell}\rightarrow\text{Comedians jokes}}\mathbb{W}_{\text{tell}\rightarrow\text{Comedians jokes}} = 4\mathbb{W}_{\text{tell}\rightarrow\text{Comedians jokes}}$ (since we want the harmony of a well-formed phrase arising from the application of two noun arguments to $\mathbb{W}_{\text{tell}}$ to reflect that it *is* a sentence upon (precise) unbinding[8]). But in DISCOG, the procedure of applying these weights to corresponding activation values at the connectionist level is precisely the act of evaluating the harmony of a symbolic structure! Given that our alternative connectionist realization is recursive on matrix-vector binding, a role cannot be applied to an argument filler unless that argument respects the API of the functional role. At 'worst', a role will return another role upon invocation in the case where an incomplete set of parameters, or fillers, are passed (much like a curried[9] function). However, a role will never compose with an incompatible filler. This is because our roles are derived from a categorical setting. In ICS, the roles suggested are merely positional, e.g

[8]Here, we are assuming a harmonic grammar of the sort $H_{\mathcal{G}} \equiv \{H(s) = H(n) \times H(n), H(n) = 2, \cdots\}$ as in (4.141).

[9]Currying is the technique of transforming a function that takes multiple arguments into a sequence of functions, each taking a single argument (partial application) and returning another function (if need be).

r_0, r_1 in the case of a binary tree. These kinds of roles only ensure the recursive nature of symbolic structures. Thus, roles in ICS fail to encode the grammar of a symbolic language beyond ordering. Conversely in DISCOG, our mappings to meaning spaces are guided by the syntax of (recursive) pregroup grammars which entails *fully grammatical* compositional rules. As such, to formulate a notion of harmony at the connectionist level in DISCOG, we do not require unbinding – we can directly evaluate the harmony of our meaning space (or meaning spaces in the case where we are dealing with a collection of *unbound*, or *incompatible*, symbolic structures). Since ICS's connectionist notion of harmony is still very much at the grammatical level, our analysis of harmony with respect to pregroup grammar is equivalent with the added value of extending to expressive meaning vectors. Hence, we arrive at the following definition of harmony at the connectionist level in DISCOG.

Proposition 4.4 (Functional Harmony of an Activation Vector)**.** *Suppose* $\mathbf{s} \in S$ *is an activation vector realizing a symbolic structure* $\mathtt{s}$ *of type* s *in a pregroup* P*. Then the functional harmony of* $\mathbf{s}$ *is defined by*

$$H(\mathbf{s}) \equiv H(\mathbf{s}) \tag{4.171}$$

$$= H(S) \tag{4.172}$$

Proof. This holds via the homomorphic mapping from P to **FVect** which assigns vector spaces to the basic types.

□

We end by noting that our definition of harmony is in fact identical to that of ICS in that checking the *dimension* of the space in which an activation vector lives so as to apply a harmony value is equivalent to checking the *type* of a grammatical object in order to apply a harmony value.

We can only see a short distance ahead, but we can see plenty there that needs to be done.

— Alan Turing

5 Conclusion and Future Work

Here, we summarize the results of this work and offer some suggestions for further investigations.

5.1 Discussion

In this thesis, we present an adaptation of the *Integrated Connectionist/Symbolic Architecture* (ICS) first introduced in [52]. ICS is a novel approach to the computational modelling of the mind that unifies connectionist and symbolic architectures via an isomorphism codified in tensor product representations. The design of ICS is an exciting development in the field of cognitive science because ICS reduces abstract cognitive functions to elementary operations that fall within the computational capabilities of neural networks, i.e. linear associators. However, the tensor product representations do admit some weaknesses. Firstly, the representational space for a concept grows in size as more elements are added to the compound. Secondly, it is only possible to compare symbolic representations with the same underlying structure. We use a category-theoretic model of meaning – the compositional distributional (DisCo) model of semantics [12] – to solve these representational issues. Namely, we build a concrete model of ICS based on the DisCo framework: compositional distributional cognition (DisCog). First, we reformulate the DisCo model of meaning as the recursive realizaton of filler/role bindings to give a connectionist account of the (ϵ, ι, and μ) maps required for information flow in DisCog. In addition, we provide simple algorithms to construct linear associators characterizing these primitive compositional operations. We then define alternative notions of filler/role binding and connectionist realization in DisCog. The result is that we produce a model of representation in higher cognition that is based on connectionist principles, while at the same time resolves the problem of

an unbounded representational space. Furthermore, our approach allows the comparison of well-formed same-type symbolic structures via realization in a (finite) *shared* meaning space. Notably, the fillers and roles we consider in DISCOG are *genuine* meaning vectors in contrast to generic ones as in ICS. Therefore, our model also improves on ICS by means of an important extension to meaning vectors. We then discuss the problem of unbinding (fillers from roles) in DISCOG and put forward a provisional (approximate) solution based on the Moore-Penrose pseudo inverse. This allows us to demonstrate how massively parallel distributed processing at the connectionist level can realize arbitrarily complex (recursive) functions at the symbolic level, of which we give an example. Lastly, we give a *functional* account of harmony in DISCOG derived from our categorical setting and show that our definition of harmony is in fact identical to that of ICS.

5.2 Further Investigations

This work represents a small step in linking the respective fields of cognitive science and quantum mechanics. An obvious extension investigating quantum properties in our model of cognition is the modelling of ambiguity with density matrices [43]. Ambiguous fillers (and roles) could be represented as *mixed* states, i.e. probability distributions over a set of possible meaning vectors. This would enhance the expressivity of DISCOG.

With respect to harmony, there is a question of how best to mend non-optimal symbolic structures. For this task, we require efficient search algorithms built on top of basic functions to manipulate cognitive representations, e.g. INSERT, PERMUTE, and DELETE.

As a final remark, we have only set up the theoretical foundations of DISCOG. Hence, a practical implementation of some sort is an evident future course of action.

References

[1] S. ABRAMSKY AND B. COECKE, *A categorical semantics of quantum protocols*, Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, (2004), pp. 415–425.

[2] ARISTOTLE, *De Anima*, Penguin Classics, 1987.

[3] E. BALKIR, *Using density matrices in a compositional distributional model of meaning*, (2014).

[4] W. BUSZKOWSKI, *Lambek grammars based on pregroups*, Logical Aspects of Computational Linguistics, (2001), pp. 95–109.

[5] J. CHENG, D. KARTSAKLIS, AND E. GREFENSTETTE, *Investigating the role of prior disambiguation in deep-learning compositional models of meaning*, NIPS 2014 Learning Semantics Workshop, (2014).

[6] S. CLARK, B. COECKE, E. GREFENSTETTE, S. PULMAN, AND M. SADRZADEH, *A quantum teleportation inspired algorithm produces sentence meaning from word meaning and grammatical structure*, (2013).

[7] S. CLARK, B. COECKE, AND M. SADRZADEH, *A compositional distributional model of meaning*, Proceedings of the Second Symposium on Quantum Interaction, (2008), pp. 133–140.

[8] S. CLARK AND S. PULMAN, *Combining symbolic and distributional models of meaning*, Proceedings of AAAI Spring Symposium on Quantum Interaction, (2007), pp. 52–55.

[9] B. COECKE, *An alternative gospel of structure: Order, composition, process*, (2013).

[10] B. COECKE, E. GREFENSTETTE, AND M. SADRZADEH, *Lambek vs. lambek: Functorial vector space semantics and string diagrams for lambek calculus*, Annals of Pure and Applied Logic, 164 (2013), pp. 1079–1100.

[11] B. COECKE, D. PAVLOVIC, AND J. VICARY, *A new description of orthogonal bases*, (2008).

[12] B. COECKE, M. SADRZADEH, AND S. CLARK, *Mathematical foundations for a compositional distributional model of meaning*, Linguistic Analysis, 36 (2011), pp. 345–384.

[13] B. COECKE AND R. SPEKKENS, *Picturing classical and quantum bayesian inference*, Synthese, 186 (2012), pp. 651–696.

[14] J. FELDMAN, *Personal communication*, 1983.

[15] J. R. FIRTH, *A synopsis of linguistic theory: 1930–1955*, Studies in Linguistic Analysis, (1957), pp. 1–32.

[16] P. GRABEN, D. PINOTSIS, D. SADDY, AND R. POTTHAST, *Language processing with dynamic fields*, Cognitive Neurodynamics, 2 (2008), pp. 79–88.

[17] E. GREFENSTETTE, *Category-theoretic quantitative compositional distributional models of natural language semantics*, (2013).

[18] ——, *Towards a formal distributional semantics: Simulating logical calculi with tensors*, Proceedings of the Second Joint Conference on Lexical and Computational Semantics, (2013).

[19] E. GREFENSTETTE, G. DINU, Y. ZHANG, M. SADRZADEH, AND M. BARONI, *Multi-step regression learning for compositional distributional semantics*, Proceedings of the 10th International Conference on Computational Semantics, (2013).

[20] E. GREFENSTETTE AND M. SADRZADEH, *Experimental support for a categorical compositional distributional model of meaning*, Proceedings of the Conference on Empirical Methods in Natural Language Processing, (2011), pp. 1394–1404.

[21] ——, *Experimenting with transitive verbs in discocat*, Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics, (2011).

[22] E. GREFENSTETTE, M. SADRZADEH, S. CLARK, B. COECKE, AND S. PULMAN, *Concrete sentence spaces for compositional distributional models of meaning*, Computing Meaning, (2014), pp. 71–86.

[23] P. HINES, *Types and forgetfulness in categorical linguistics and quantum mechanics*, Quantum Physics and Linguistics: A Compositional, Diagrammatic Course, (2013).

[24] N. KALCHBRENNER AND P. BLUNSOM, *Recurrent convolutional neural networks for discourse compositionality*, Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality, (2013).

[25] N. KALCHBRENNER, E. GREFENSTETTE, AND P. BLUNSOM, *A convolutional neural network for modelling sentences*, Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, (2014).

[26] D. KARTSAKLIS, *Compositional operators in distributional semantics*, Springer Science Reviews, (2014).

[27] D. KARTSAKLIS, N. KALCHBRENNER, AND M. SADRZADEH, *Resolving lexical ambiguity in tensor regression models of meaning*, Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), (2014), pp. 212–217.

[28] D. KARTSAKLIS AND M. SADRZADEH, *A study of entanglement in a categorical framework for natural language*, Proceedings of the 11th Workshop on Quantum Physics and Logic, (2014).

[29] D. KARTSAKLIS, M. SADRZADEH, AND S. PULMAN, *A unified sentence space for categorical distributional-compositional semantics: Theory and experiments*, Proceedings of 24th International Conference on Computational Linguistics, (2012), pp. 549–558.

[30] D. KARTSAKLIS, M. SADRZADEH, S. PULMAN, AND B. COECKE, *Reasoning about meaning in natural language with compact closed categories and frobenius algebras*, Logic and Algebraic Structures in Quantum Computing and Information, (2013).

[31] P. KÖNIG, K. KÜHNBERGER, AND T. KIETZMANN, *A unifying approach to high- and low-level cognition*, Models, Simulations, and the Reduction of Complexity, (2014), pp. 117–141.

[32] J. LAMBEK, *Type grammar revisited*, Logical Aspects of Computational Linguistics, 1582 (1999), pp. 1–27.

[33] ——, *From Word to Sentence: A Computational Algebraic Approach to Grammar*, Polimetrica, 2008.

[34] ——, *Compact monoidal categories from linguistics to physics*, New Structures for Physics, (2010), pp. 451–469.

[35] A. LAUB, *Notes on Moore-Penrose Pseudo Inverse*, University of California, Los Angeles, Los Angeles, CA, May 2012.
Available: http://www.math.ucla.edu/ laub/33a.2.12s/mppseudoinverse.pdf.

[36] M. LEWIS AND B. COECKE, *A compositional explanation of the 'pet fish' problem*, (2014).

[37] J. MAILLARD, S. CLARK, AND E. GREFENSTETTE, *A type-driven tensor-based semantics for ccg*, EACL 2014 Type Theory and Natural Language Semantics Workshop, (2014).

[38] D. MILAJEVS, D. KARTSAKLIS, M. SADRZADEH, AND M. PURVER, *Evaluating neural word representations in tensor-based compositional settings*, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, (2014).

[39] R. MONTAGUE, *Universal grammar*, Theoria, 36 (1970), pp. 373–398.

[40] ——, *English as a formal language*, in Formal Philosophy: Selected Papers of Richard Montague, Yale University Press, 1974, pp. 188–222.

[41] J. PATER, *Review of Smolensky and Legendre: The Harmonic Mind*, University of Massachusetts, Amherst, Amherst, MA, April 2009.
Available: http://roa.rutgers.edu/files/983-0708/983-PATER-0-0.PDF.

[42] R. PIEDELEU, *Ambiguity in categorical models of meanings*, (2014).

[43] R. PIEDELEU, D. KARTSAKLIS, B. COECKE, AND M. SADRZADEH, *Modelling ambiguity with density matrices in compositional distributional models of meanings*, Advances in Distributional Semantics, (2015).

[44] ——, *Open system categorical quantum semantics in natural language processing*, Proceedings of the 6th Conference on Algebra and Coalgebra in Computer Science, (2015).

[45] PLATO, *Phaedo*, Oxford Paperbacks, 2009.

[46] E. POTHOS AND J. TRUEBLOOD, *Structured representations in a quantum probability model of similarity*, Journal of Mathematical Psychology, 64 (2015), pp. 35–43.

[47] M. SADRZADEH, S. CLARK, AND B. COECKE, *The frobenius anatomy of word meanings i: Subject and object relative pronouns*, Journal of Logic and Computation, 23 (2013), pp. 1293–1317.

[48] ——, *The frobenius anatomy of word meanings ii: Possessive relative pronouns*, Journal of Logic and Computation, (2014).

[49] M. SADRZADEH AND E. GREFENSTETTE, *A compositional distributional semantics, two concrete constructions, and some experimental evaluations*, Lecture Notes in Computer Science, 7052 (2011), pp. 35–47.

[50] H. SCHÜTZE, *Automatic word sense discrimination*, Computational Linguistics, 24 (1998), pp. 97–123.

[51] P. SELINGER, *A survey of graphical languages for monoidal categories*, New Structures for Physics, (2011), pp. 289–355.

[52] P. SMOLENSKY AND G. LEGENDRE, *The Harmonic Mind: From Neural Computation to Optimality-Theoretic Grammar (Volume 1: Cognitive Architecture)*, MIT Press, 2005.

[53] P. TURNEY AND P. PANTEL, *From frequency to meaning: Vector space models of semantics*, Journal of Artificial Intelligence Research, 37 (2010), pp. 141–188.

[54] V. ZABOTKINA AND E. BOYARSKAYA, *Sense disambiguation in polysemous words: Cognitive perspective*, Psychology in Russia: State of the Art, 6 (2013), pp. 60–67.