Autonomization of monoidal categories

Antonin Delpeuch

Department of Computer Science, University of Oxford

May 9, 2019

We show that contrary to common belief in the DisCoCat community, a monoidal category is all that is needed to define a categorical compositional model of natural language. This relies on a construction which freely adds adjoints to a monoidal category. In the case of distributional semantics, this broadens the range of available models, to include non-linear maps and cartesian products for instance. We illustrate the applications of this principle to various distributional models of meaning.

1 Introduction

The DisCoCat model [Clark et al., 2008, Coecke et al., 2010] applies category theory to linguistics. The idea is to view the type-logical representation of grammatical structure [Lambek, 2008] as a morphism in a compact closed category, and use this morphism to combine individual word meanings into sentence representations. This gives an inherently compositional template for models of meaning which respect the grammatical structure by design. The flagship application of this idea is its interpretation it in vector-based categories, as it makes it applicable to distributional semantics (hence called the **Dis**tributional **Co**mpositional **Cat**egorical model). Distributional semantics represent word meanings by computing co-occurence statistics of words across large corpora [Schütze, 1998], or more recently by training neural networks to learn word embeddings [Mikolov et al., 2013]. The task of generalizing these representations from words to larger syntactical units such as sentences is a challenging task, and the promise of the DisCoCat approach is to tackle it with a categorical viewpoint.

The mantra of DisCoCat is that a model of meaning is given by a compact closed category and a choice of word meanings in it [Coecke et al., 2010]. This definition is flexible in that it encompasses both logical models of meaning and distributional ones, depending on the concrete category chosen [Clark et al., 2008]. However, the requirement to define a compact closed category is still a restricting one, as it rules out a wide range of categories where objects do not always have adjoints. The general consensus

Antonin Delpeuch: antonin.delpeuch@cs.ox.ac.uk

in the community was that when it comes to vector-based models of meanings, DisCoCat requires the semantics to be assembled out of linear maps only [Wijnholds and Sadrzadeh, 2018, Lewis, 2019]. We show how to lift this restriction, by only requiring the semantic category to be monoidal, and freely adding the caps and cups required for the interpretation of the grammatical structure [Delpeuch, 2014]. We briefly present these results in Section 2.

The bulk of this piece consists in showing how various models of meaning formulated outside the DisCoCat framework can actually be recast in it, thanks to the free addition of adjoints. Our aim is to understand why the DisCoCat community has pushed itself in the narrow corner of linear models for such a long time, in a context where the wider linguistics community has overwhelmingly embraced non-linear models. By conducting this analysis, our hope is to encourage the wider applied category theory community to confront their models with the actual practitioners in the field targeted, and challenge any categorical dogma with their feedback.

2 Free autonomous categories

We recall a few useful definitions.

Definition 1. A ***monoidal category*** is a category $\mathcal{C}$ equipped with a symmetric bifunctor $\_ \otimes \_ : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$. This operation is furthermore required to have a unit $I \in \mathcal{C}$ and to be naturally associative.

A monoidal category is symmetric when its monoidal product is.

Definition 2. An ***autonomous*** (or rigid) ***category*** is a monoidal category such that any object A has left and right adjoints, meaning that there are morphisms $\epsilon_l : A^l \otimes A \to 1$, $\epsilon_r : A \otimes A^l \to 1$, $\eta_l : 1 \to A \otimes A^l$ and $\eta_r : 1 \to A^r \otimes A$ satisfying equations.

In a symmetric monoidal category, left and right adjoints are isomorphic. An autonomous category that is also symmetric is called a **compact closed category**. An example of such category is $(\mathbf{Vect}, \otimes, I)$, the category of finite-dimensional vector spaces and linear maps, with the tensor product as monoidal structure.

Definition 3. A ***cartesian category*** is a category with all finite products. In other words it is a symmetric monoidal category with a natural family of copying and discarding maps satisfying the usual laws.

An example of cartesian category is $(\mathbf{Vect}, \oplus, O)$, the category of finite-dimensional vector spaces and linear maps, with the direct sum as monoidal structure. Another canonical example is $(\mathbf{Set}, \times, \{*\})$, the category of sets and functions equipped with the (set-theoretic) cartesian product.

Proposition 1. If a cartesian category is also autonomous for the same monoidal structure, then every object is isomorphic to the monoidal unit.

Pat   grows   delicious   kiwis

$$n \qquad n^r \ s \ n^l \qquad n \ n^l \qquad n \qquad = \qquad s$$
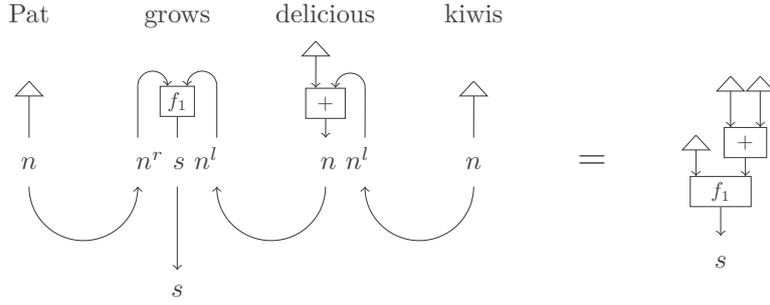
Figure 1: Eliminating free caps and cups from a sentence meaning

In other words, cartesian and autonomous structures are incompatible.

Theorem 1. Any monoidal category $(\mathcal{C}, \_\otimes\_, I)$ gives rise to a free autonomous category $(L(\mathcal{C}), (\_, \_), ())$. The embedding functor $F : \mathcal{C} \to L(\mathcal{C})$ is strong monoidal and faithful.

This means that it is possible to freely add caps and cups to any monoidal category. To emphasize the different nature of the monoidal structures on both categories, we use different notations for them. They are related by the embedding functor F which is strong monoidal. Informally, the category $L(\mathcal{C})$ is defined by taking formal string diagrams annotated with morphisms of $\mathcal{C}$. This result seems to known as folklore in some communities. For the sake of rigour we provide a detailed construction in Appendix A.

Morphisms in $L(\mathcal{C})$ are of little use for semantics as they are formal objects. Luckily, in the particular case where the domain and codomain are objects of $\mathcal{C}$, we can eliminate the formal caps and cups introduced to recover a morphism in $\mathcal{C}$. This is an equivalent of the normalization property of Preller and Lambek [2007] for our free construction.

Theorem 2. The functor F is full.

Proof. Let $f \in L\mathcal{C}(F(A), F(B))$ with $A, B \in \mathcal{C}$, we show that $f \in F(\mathcal{C}(A, B))$. If no formal caps and cups occur in f, then it can be expressed as a vertical and horizontal composition of generator morphisms from $\mathcal{C}$, so it belongs to $F(\mathcal{C}(A, B))$. Therefore we only need to show that all caps and cups can be eliminated from a diagram whose boundaries do not contain any adjoints. As no generator contains adjoints in their domain or codomain, the winding number of any wire in the diagram f is null. For any unit occuring in f we can find a matching counit on the same wire, such that the unit and counit can be cancelled together. This elimination property is shown as Lemma 3.12 in Mimram [2014]. □

What fullness of F means is that any morphism in $\mathcal{C}^*$ between objects coming from $\mathcal{C}$ actually comes from $\mathcal{C}$ too. Because sentence meanings in DisCoCat are interpreted as elements of the sentence space S, they are therefore morphisms in $\mathcal{C}^*([I], [S])$. By fullness this means that they correspond to morphisms $\mathcal{C}(I, S)$ (and a unique one by

faithfulness). This means that sentence meanings are not formal objects: they belong to the original semantic monoidal category.

The use of $\mathcal{C}^*$ as a model of meaning is illustrated in Figure 1. The recipe is simple:

- define word meanings, using formal caps to produce elements of the types dictated by the grammar;

- compose word meanings with the formal cups determined by the type reduction witnessing the grammaticality of the sentence;

- eliminate the formal caps and cups in the resulting diagram. Proposition **??** guarantees that all formal caps and cups can be eliminated.

- you obtain an element of the sentence space in the original category $\mathcal{C}$, which is the representation of the sentence.

In the sequel, we illustrate the concrete use of this result, showing how it can expand the range of models of meaning which can be formulated in the DisCoCat framework. We also argue that it weakens the case for $\mathbf{Vect}_\otimes$, which has so far been considered as the canonical (if not only) semantic category suited for distributional compositional semantics.

3   Some early evidence against tensors

Let us first recall how the DisCoCat model and its tensor-based interpretation came into existence. The interest in the tensor product for distributional semantics was initially suggested by Clark and Pulman [2007], inspired by ideas from cognitive science. The idea was that tensors can account for correlations between the meanings of some words. This was demonstrated by a thought experiment called the *pet fish problem* [Aerts and Gabora, 2005]. The idea was that the conjunction of two concepts is not always best represented by the independent superposition of both concepts: there is some interaction between the two concepts. It was reported that subjects found the image of a *guppy* was neither a prototypical image of *pet* nor that of a *fish*. Inspired by quantum mechanics, it was proposed to represent concepts in Hilbert space, such that state entanglement could be used to model this phenomenon.

This quantum-inspired model of meaning became all the more motivated when the connection to compact closed categories was made by Clark et al. [2008], Coecke et al. [2010]. Indeed, vector spaces form a compact closed category if the tensor product is used as monoidal structure, but this fails for other natural structures such as the direct sum, as explained in the previous section. All this evidence, coming simultaneously and independently from cognitive science and category theory, seemed to indicate in an unambiguous way that $\mathbf{Vect}_\otimes$ was the natural category to develop distributional compositional semantics.

A few difficulties with this proposal became soon clear, and the first one was the high dimensionality of the word representations involved. Indeed, the number of parameters of word representations grows exponentially with the length of its grammatical type. This means for instance that ditransitive verbs such as *give*, which are applied to a subject and two objects, would be represented by $n^3 s$ parameters, where n is the dimensionality of the sentence space and n that of the noun space. Paradoxically, this prohibitive cost was not seen as a fundamental problem of the model – it somehow justified the absence of large-scale benchmarks against competing approaches. It also suggested that the model was more expressive or powerful, having more parameters than other models: this expressivity was only waiting for the right hardware (such as quantum devices) to be unleashed [Zeng and Coecke, 2016].

However, various attempts were made to learn the high-order word representations suggested by the framework [Grefenstette et al., 2013, Kartsaklis and Sadrzadeh, 2014, Grefenstette and Sadrzadeh, 2015]. Still, these experiments were only conducted for simple syntactic structures such as *adjective-noun* or *subject-verb-object* patterns, far from the coverage of arbitrary text generally expected in the field natural language processing.

One of these approaches was that of Polajnar et al. [2014], where it was proposed to restrict verb meanings to particular shapes, reducing the number of parameters to learn. The intention was to stick to the DisCoCat framework using $\mathbf{Vect}_\otimes$, but impose further restrictions on the semantic spaces and word meanings to obtain more tractable and effective models. Four approaches to model transitive verbs in *subject-verb-object* sentences were compared:

(i) **Tensor**: the standard, unrestricted representation of transitive verbs as tensors of type $N \otimes N \otimes S$, where N and S are sentence and noun spaces. The space S is two-dimensional, the two truth values corresponding to orthogonal basis vectors.

(ii) **KKMat**: the same approach, but collapsing S to a single dimension, the truth value being represented by the norm of the vector.

(iii) **SKMat**: verbs are represented matrices in $N \otimes S$ (where S is two dimensional again). They are multiplied by the object vector seen as a diagonal matrix, and then contracted with the subject vector.

(iv) **2Mat**: verbs are represented by two $N \otimes S$ tensors. They are multiplied with the subject and object respectively to obtain two vectors in S, which are finally concatenated into a four-dimensional vector.

(a) **Tensor**, the most generic form for a transitive verb

(b) **KKMat**, assuming that $S = I$

(c) **SKMat**, where the circular node is the Frobenius copy map for the canonical basis

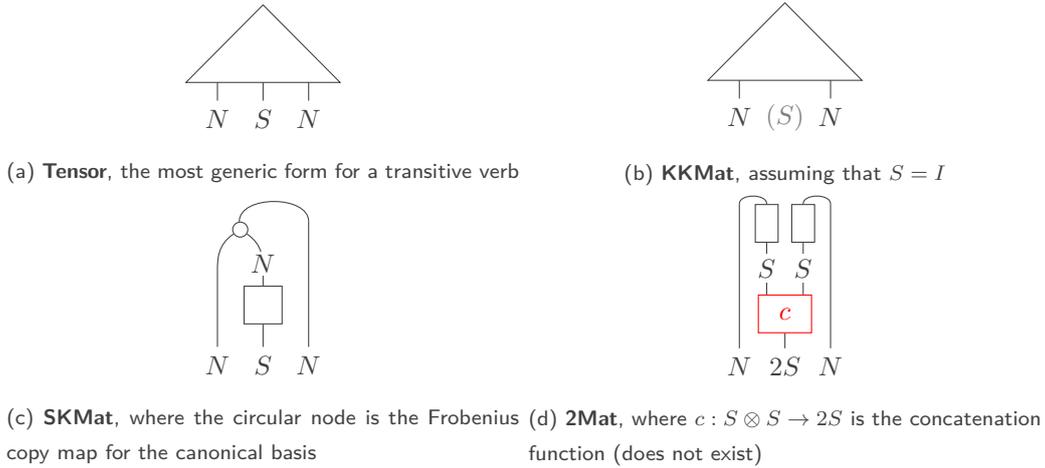(d) **2Mat**, where $c : S \otimes S \to 2S$ is the concatenation function (does not exist)

Figure 2: The four approaches to dimensionality reduction studied by Polajnar et al. [2014]

These four approaches, represented as string diagrams in Figure 2, were evaluated on a task which consists in estimating the plausibility of a candidate sentence and the best model overall was **2Mat**. What the authors did not realize however was that unlike the three other approaches, **2Mat** cannot be formulated in $\mathbf{Vect}_\otimes$ since it involves concatenating vectors, which is not a bilinear operation. In $\mathbf{Vect}_\otimes$, any morphism taking two arguments as inputs is necessarily bilinear in them, since it is a linear map from the tensor product of their spaces. So there is no function $c : S \otimes S \to S \oplus S$ such that $c(a \otimes b) = (a, b)$ for all $a, b \in S$. The fact that the best performing approach to dimensionality reduction in DisCoCat was a model that actually violated the assumptions of DisCoCat could have been seen as a sign that the assumptions of the model should have been revisited, but it actually went largely unnoticed in the community.

The reason why the **2Mat** model was allowed to slip out of $\mathbf{Vect}_\otimes$ is that it was formulated in terms of function application rather than tensor contraction. These two views are equivalent in a compact closed category such as $\mathbf{Vect}_\otimes$ but only when the appropriate monoidal product $\otimes$ is used to combine the arguments together. We argue that thinking in terms of function application to design models of meaning is much more intuitive than using tensor contraction, especially to linguists not trained in quantum physics.

All the models proposed by Polajnar et al. [2014], including **2Mat**, can be recast as DisCoCat models in $\mathbf{Set}^*_\times$, the free autonomous category on $\mathbf{Set}_\times$, the category of sets and functions with the cartesian product. In this setting, there are no tensors to contract: meaning composition can only be thought of in terms of function application. In fact, the models of meaning which can be formulated in $\mathbf{Set}^*_\times$ are precisely all the functional models directed by the grammar.

4   Further evidence against linearity

Beyond the requirement to use tensors to pair up the arguments of any morphism with multiple inputs, $\mathbf{Vect}_\otimes$ also requires that morphisms to be linear. This requirement has not been challenged much in the community either. This is perhaps because linearity assumptions are pervasive in many fields of science; non-linear models can often be reduced to linear ones via some transformation. In the particular context of $\mathbf{Vect}_\otimes$ however, there is little room for such adaptations. In this section, we show why this might be a problem in its own right.

In linear algebra, an affine map is a map $f : x \mapsto Ax + b$, where A is a linear map and b is a constant vector. Affine maps are so similar to linear maps that the two notions are often conflated: $Ax + b$ is coloquially called a linear combination of the elements of x. In fact, any affine map $f : X \to Y$ can be seen as a linear map $f' : X + \mathbf{1} \to Y$, where $\mathbf{1}$ is the one-dimensional vector space for the same field: $f(x) = f'(x, 1)$. With this observation, it is possible to adapt the matrix calculus of linear maps to work for affine maps.

However, this trick can sadly not be used to generalize $\mathbf{Vect}_\otimes$ to affine maps. The product $(A + \mathbf{1}) \otimes (B + \mathbf{1})$ is not equal to $(A \otimes B) + \mathbf{1}$ for non-null spaces and there does not seem to be a natural generalization of bilinearity to allow for constants. Affine maps and tensor products just do not mix up well. This shows how stringent the requirement to use $\mathbf{Vect}_\otimes$ is: even the seemingly innocuous addition of a constant breaks the machinery down.

This vow to strict linearity is of course problematic in a context where neural networks have become the machine learning models of choice in natural language processing, as these models crucially rely on nonlinearities to function. We will review the possible interactions between DisCoCat and neural networks in the next section, but let us first give linearity the benefit of the doubt: it could be that non-linearities are needed to learn word vectors, but the resulting vectors could then be composed linearly when deriving the representation of larger text units [Milajevs et al., 2014, Sadrzadeh, 2017].

One of the seminal results that popularized neural word embeddings in linguistics was that of Mikolov et al. [2013]. After learning word embeddings from text with a simple model that is trained to guess a missing word in a fixed-length textual context, they observe that the learned vectors satisfy some promising semantic properties. The classical example of such a property is the fact that $\overrightarrow{King} - \overrightarrow{Man} + \overrightarrow{Woman}$ gives a vector whose nearest neighbour is $\overrightarrow{Queen}$. This sort of relation between vectors, which arises from the training process without being enforced directly by the learning objective, invites to model predicates as additive functions.

This insight proved successful in a related problem, that of learning embeddings for entities in a knowledge graph. A knowledge graph is essentially a directed multigraph

where edges are annotated with properties. These edges are seen as *triples* formed by their source, predicate and target, each encoding a fact about the world, such as the triple (`Cantaloupe_Island_(song)`, `composer`, `Herbie_Hancock`) for instance. Graph embeddings associate to each entity in the knowledge graph a vector in a finite-dimensional vector space, computed solely from the knowledge graph itself. Such embeddings are then useful to tackle various information retrieval tasks, such as completing the knowledge graph by adding missing triples. One of the popular approaches to compute such embeddings is the **TransE** model [Bordes et al., 2013], which consists in optimizing the learning objective that $v(o) - v(s) \simeq v(p)$ for each triple (s, p, o) (subject – predicate – object).

Whether word vectors are learned from text or from a knowledge graph, it is therefore tempting to model predicates in a distributional compositional model by such additive functions. For instance, the representation of the *royal* adjective could be a function adding $\overrightarrow{Queen} - \overrightarrow{Woman}$ to its argument. Sadly, this is again not a linear function but an affine one: it cannot be implemented in $\mathbf{Vec}_\otimes$. Again, it is a perfectly valid semantic representation in $\mathbf{Set}_\times^*$.

5   Recasting convolutional models in DisCoCat

Neural networks are popular to learn word vectors, but the dominant architectures to learn these word vectors ignore grammar entirely: the networks are simply trained to predict words in a text from their surrounding context, regardless of their grammatical function [Mikolov et al., 2013]. Some of these models, such as the ones designed to translate text between languages [Sutskever et al., 2014], make use of vector representations for larger units of texts such as sentences: however, these vectors are rarely designed to encode the full meaning of the textual unit, as the translation to the target language relies on *attention mechanisms* which rely on the individual vectors as well as the combined representation to produce the desired output [Bahdanau et al., 2014]. As such, they do not fully address the problem of combining distributional representations.

However, neural networks have also been applied in syntax-aware approaches which construct sentence representations from word representations. In Socher et al. [2013], a small neural network is applied recursively along the syntax tree of the sentence to derive a vector representation of each node from that of its children. Lewis [2019] proposed to recast this approach in the DisCoCat framework, which would give a similar model driven from a pregroup-induced representation of the grammar instead. But doing so required them to exchange the neural network for a tensor in order to formulate the model in $\mathbf{Vect}_\otimes$. Choosing to formulate it in $\mathbf{Set}_\times^*$ instead lets us keep the original, non-linear convolutional unit. In effect, this just amounts to swapping the context-free grammar used by Socher et al. [2013] for a type-driven grammar such as a

pregroup grammar. In fact, a similar proposal was formulated by Krishnamurthy and Mitchell [2013] earlier, using Combinatory Categorial Grammar instead of pregroup grammar. This proposal also falls into the scope of DisCoCat, when implemented in $\mathbf{Set}^*_\times$.

6 Conclusion

We have shown that contrary to common belief in the DisCoCat community, a monoidal category is all that is needed to define a compositional model of meaning. In the case of distributional semantics, this broadens the range of available models to include non-linear maps and cartesian products. We hope this will encourage the community to experiment with alternate models, which should be more amenable to competing with the state of the art in natural language processing.

7 Acknowledgements

References

Diederik Aerts and Liane Gabora. A Theory of Concepts and Their Combinations II: A Hilbert Space Representation. *Kybernetes*, 34(1/2):192–221, January 2005. ISSN 0368-492X. DOI: 10.1108/03684920510575807.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*, September 2014.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, page 9, 2013.

Stephen Clark and Stephen Pulman. Combining Symbolic and Distributional Models of Meaning. In *AAAI Spring Symposium: Quantum Interaction*, pages 52–55, 2007.

Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. A compositional distributional model of meaning. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*, pages 133–140, 2008.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical Foundations for a Compositional Distributional Model of Meaning. *CoRR*, abs/1003.4394, 2010.

Antonin Delpeuch. Autonomization of Monoidal Categories. *arXiv:1411.3827 [cs, math]*, November 2014.

Edward Grefenstette and Mehrnoosh Sadrzadeh. Concrete Models and Empirical Evaluations for the Categorical Compositional Distributional Model of Meaning. *Computational Linguistics*, 41(1):71–118, February 2015. ISSN 0891–2017. DOI: 10.1162/COLI_a_00209.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. Multi-Step Regression Learning for Compositional Distributional Semantics. *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, 2013.

André Joyal and Ross Street. Planar diagrams and tensor algebra. Unpublished manuscript, available from Ross Street's website, September 1988.

André Joyal and Ross Street. The geometry of tensor calculus, I. *Advances in Mathematics*, 88(1):55–112, 1991.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. A Study of Entanglement in a Categorical Framework of Natural Language. In *Proceedings of the 11th Workshop on Quantum Physics and Logic (QPL)*, Kyoto, Japan, June 2014. DOI: 10.4204/eptcs.172.17.

Jayant Krishnamurthy and Tom M Mitchell. Vector space semantic parsing: A framework for compositional vector space models. *ACL 2013*, page 1, 2013.

J Lambek. Pregroup Grammars and Chomsky's Earliest Examples. *Journal of Logic, Language and Information*, 17(2):141–160, 2008.

Martha Lewis. Compositionality for Recursive Neural Networks. *CoRR*, abs/1901.10723, 2019.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. Evaluating Neural Word Representations in Tensor-Based Compositional Settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719, Doha, Qatar, 2014. Association for Computational Linguistics. DOI: 10.3115/v1/D14–1079.

Samuel Mimram. Towards 3-Dimensional Rewriting Theory. *Logical Methods in Computer Science*, 10(2), April 2014. ISSN 18605974. DOI: 10.2168/LMCS–10(2:1)2014.

Tamara Polajnar, Luana Fagarasan, and Stephen Clark. Reducing Dimensions of Tensors in Type-Driven Distributional Semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1036–1046, Doha, Qatar, 2014. Association for Computational Linguistics. DOI: 10.3115/v1/D14–1111.

Anne Preller and Joachim Lambek. Free Compact 2-categories. *Mathematical. Structures in Comp. Sci.*, 17(2):309–340, April 2007. ISSN 0960–1295. DOI: 10.1017/S0960129506005901.

Mehrnoosh Sadrzadeh. Quantization, Frobenius and Bi Algebras from the Categorical Framework of Quantum Mechanics to Natural Language Semantics. *Frontiers in Physics*, 5, 2017. ISSN 2296-424X. DOI: 10.3389/fphy.2017.00018.

Hinrich Schütze. Automatic word sense discrimination. *Comput. Linguist.*, 24(1):97–123, March 1998. ISSN 0891-2017.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. *arXiv:1409.3215 [cs]*, September 2014.

Gijs Wijnholds and Mehrnoosh Sadrzadeh. Classical Copying versus Quantum Entanglement in Natural Language: The Case of VP-ellipsis. *arXiv preprint arXiv:1811.03276*, 2018.

William Zeng and Bob Coecke. Quantum Algorithms for Compositional Natural Language Processing. *Electronic Proceedings in Theoretical Computer Science*, 221:67–75, August 2016. ISSN 2075-2180. DOI: 10.4204/EPTCS.221.8.

A   Constructing autonomous categories

A.1   Idea of the construction

Let $\mathcal{C}$ be a strict monoidal category. Our goal is to embed $\mathcal{C}$ in a "larger" category, $L(\mathcal{C})$, which will be autonomous. The embedding has to be functorial, so that the original composition operations are retained.

The ϵ and η maps will be purely formal, which means that they will have no interpretation in the original category. Our approach to define them consists in taking diagrams seriously: the arrows of our autonomous category will be diagrams. To do so, we adapt the definitions of Joyal and Street [1988], who defined the diagrams for autonomous categories and proved their soundness and completeness. They assume for simplicity that the links in the diagrams are piecewise linear and we follow their choice.

Our construction is similar to that of Preller and Lambek [2007] who defined the autonomous category freely generated by a category $\mathcal{C}$, not assumed to be monoidal. Taking into account the monoidal structure of $\mathcal{C}$ construction is important for the linguistic applications mentioned in Section ??, as we need some compatibility between the initial monoidal structure in $\mathcal{C}$ and the monoidal structure of the larger category $L(\mathcal{C})$.

In this section, we summarize the definitions from Joyal and Street [1988] needed to define our category $L(\mathcal{C})$. Our goal is not to give the most general definition of topological graphs, but only to define precisely the objects that we will manipulate.

A **graph** Γ is a compact subset of $\mathbb{R}^2$ together with a subset $\Gamma_0 \subseteq \Gamma$ such that

(i) Γ_0 is discrete and finite, its elements are called **nodes**;

(ii) $\Gamma - \Gamma_0$ has a finite number of connected components called **1–cells** and each of them is homeomorphic to an open interval.

We denote by $[p;q]$ the segment connecting two points $p, q \in \mathbb{R}^2$. For all tuples of points $t = (p_0, \ldots, p_n)$ we define $[p_0, \ldots, p_n] = [p_0, p_1] \cup [p_1, p_2] \cup \cdots \cup [p_{n-1}, p_n]$. The latter is called a **piecewise linear segment**. The tuple t is called **reduced** when there is no $0 < i < n$ such that $p_i \in [p_{i-1}, p_{i+1}]$. In this case, the p_i for $0 < i < n$ are called **singular points**. A parametrization of a reduced piecewise linear segment $\gamma : [0;1] \to [p_0, \ldots, p_n]$ such that $\gamma(0) = p_0$ and $\gamma(1) = p_n$ is called a **piecewise linear curve**. The **initial** (respectively **terminal**) segment of such a γ is $[p_0, p_1]$ (respectively $[p_{n-1}, p_n]$). We denote by $\bar{\gamma}$ the reversed parametrization: $\bar{\gamma}(t) = \gamma(1 - t)$.

A **piecewise linear graph** Γ is a graph where the closure of any 1–cell is a piecewise linear segment, and such that no initial or terminal segment is horizontal. The **edges** of such a graph are the parametrizations of these closures, identified up to monotonous reparametrization (hence a 1–cell gives rise to two edges, γ and $\bar{\gamma}$). The set of edges is denoted by $\mathrm{Edges}(\Gamma)$. In the rest of this paper, all graphs are assumed to be piecewise linear.

Let γ be an edge and $x < y \in [0;1]$ be preimages of consecutive singular points of $\gamma([0;1])$. The segment $[\gamma(x), \gamma(y)]$ is **directed top** (respectively **bottom**) when the second coordinate of $\gamma(y)$ is greater (respectively smaller) than that of $\gamma(x)$. The last requirement of the definition of a piecewise linear graph implies that initial and terminal segments of edges are either directed top or bottom. This allows us to define the **inputs** of a node x as the set of edges γ such that $\gamma(1) = x$ and the terminal segment of γ is directed bottom. Similarly, the **outputs** of x are the edges γ such that $\gamma(0) = x$ and the initial segment of γ is directed bottom.

A graph Γ is **between slices** a and b, where $a < b$ are reals, when $\Gamma \subset \mathbb{R} \times [a;b]$, and such that every node in $\mathbb{R} \times \{a\}$ (respectively b) has one input and no output (respectively one output and no input). These nodes included in $\mathbb{R} \times \{a, b\}$ are called **outer nodes** and the others are **inner nodes**. The set of inner nodes of a graph Γ is denoted by $\mathrm{Nodes}(\Gamma)$. The reason for this notation is that the outer nodes will not represent morphisms but simply "gates", i.e. inputs and outputs of the diagram. The outer nodes included in $\mathbb{R} \times \{a\}$ are called **lower outer nodes** and the other outer nodes are called **upper outer nodes**. A **regular slice** is a $c \in [a, b]$ such that $\Gamma_0 \cap \mathbb{R} \times \{c\} = \emptyset$. A **unit graph** is a graph included in $[0, 1]^2$ and between slices 0 and 1.

Finally, we need to define the turning number $\rho(\gamma)$ of an edge γ. Informally, this is the number of half-turns of the edge in the direct orientation, minus the number of half-turns in the indirect orientation. We invite the interested reader to consult Joyal and Street [1988] for a rigorous definition. The following examples should be enough to grasp the idea:

$$\gamma =$$

$$\rho(\gamma) = \qquad 1 \qquad\qquad 1 \qquad\qquad -1 \qquad\qquad 2 \qquad\qquad 0$$
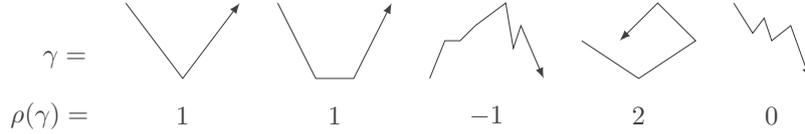
Definition 4. A ***yankable graph*** is a graph between slices a and b such that for every edge γ between two inner nodes, $\rho(\gamma) = 0$.

The reason for this additional requirement $\rho(\gamma) = 0$ is that we will attribute a morphism of $\mathcal{C}$ to each node in Section A.4. Informally, as the domain and the codomain of such morphisms cannot contain *adjoints*, it is necessary that the links between them can be *yanked* to a straight line.

Here are a few examples of yankable and not yankable graphs. The edges that make the graphs not yankable are drawn with dotted lines.
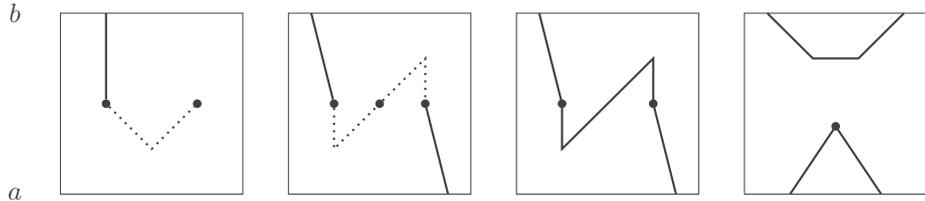
A **deformation of graphs** Γ to Γ' is a regular deformation of polarised graphs between Γ and Γ', as defined in Joyal and Street [1988]. As it preserves the turning number of edges, it preserves yankable graphs.

### A.3   Occurrences and replacement

We define what an occurrence of a graph G_1 in a graph Γ is, and what the substitution of G^1 by G^2 in Γ is. This will be useful to define an equivalence relation on graphs, which will be required to define the autonomous category $L(\mathcal{C})$ properly.

This notion is not needed to obtain the soundness and correctness results of Joyal and Street [1988], so one could wonder why we introduce it while dealing with the same objects. The reason is that the autonomous category we are constructing cannot be completely free, as we have to retain the equalities holding in the original monoidal category. This will enable us to define a functorial embedding in Section B.1.

Let $t = (a, b, c, d)$ with $a < b$ and $c < d$ be reals. We define an homeomorphism $\phi_t : [0,1]^2 \to [a,b] \times [c,d]$ by $\phi_t(x,y) = (a + x(b-a), c + y(d-c))$. Let G^1 and G^2 be unit

graphs, such that the outer nodes of G^1 and G^2 are the same. An **occurrence** of G^1 in a graph $\Gamma \subset \mathbb{R}^2$ is a quadruplet of reals $t = (a, b, c, d)$, $a < b$ and $c < d$, such that $\phi_t(G^1) = \Gamma \cap ([a, b] \times [c, d])$, and such that no node of Γ is included in the boundary of $[a, b] \times [c, d]$. We define $\Gamma[G^1 := G^2]_t = (\Gamma - \phi_t(G^1)) \cup \phi_t(G^2)$, with nodes $(\Gamma_0 - \phi_t(G_0^1)) \cup \phi_t(G_0^2)$.
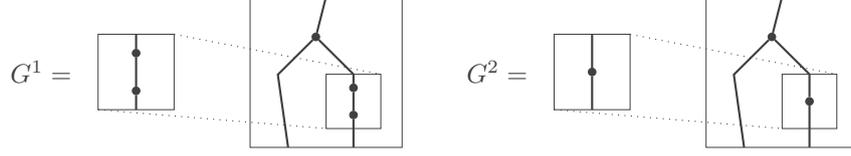
One can check that $\Gamma[G_1 := G_2]_t$ is a piecewise linear graph. However, it is not yankable in general. We will get this guarantee with the notion of valued graphs introduced in the next section.

A.4 Valued graphs

In this section, we add *valuations* to the objects introduced in the previous section. This consists in labelling the nodes and the edges with objects and arrows from a category, in a consistent way.

Definition 5. A $\mathcal{C}$-**valued graph** is a yankable graph Γ with functions

$$v_0 : \mathrm{Nodes}(\Gamma) \to \mathrm{Mor}(\mathcal{C}) \qquad\qquad v_1 : \mathrm{Edges}(\Gamma) \to \mathrm{Ob}(\mathcal{C}) \times \mathbb{Z}$$

such that:

(i) $\forall \gamma \in \mathrm{Edges}(\Gamma), v_1(\bar{\gamma}) = (A, n + \rho(\gamma))$ where $(A, n) = v_1(\gamma)$

(ii) $\forall x \in \mathrm{Nodes}(\Gamma), \forall \gamma \in \mathrm{Edges}(\Gamma)$

- if $\gamma(0) = x$ then $v_1(\gamma) = (A, 0)$ for some $A \in \mathrm{Ob}(\mathcal{C})$
- if $\gamma(1) = x$ then $v_1(\bar{\gamma}) = (A, 0)$ for some $A \in \mathrm{Ob}(\mathcal{C})$

(iii) $\forall x \in \mathrm{Nodes}(\Gamma), v_0(x) : v_1(\bar{\gamma}_1) \otimes \cdots \otimes v_1(\bar{\gamma}_p) \to v_1(\delta_1) \otimes \cdots \otimes v_1(\delta_q)$

where $\gamma_1, \ldots, \gamma_p$ and $\delta_1, \ldots, \delta_q$ are the ordered lists of the input and output edges of x, and where $v_1^e(\gamma) = (A, 0)$ is identified with A for simplicity.

Informally, the value $v_1(\gamma)$ represents the domain of the edge γ, and $v_1(\bar{\gamma})$ represents its codomain. The condition (i) states the relation between the two. Note that a valued graph is always yankable, as the rotation number of an edge between two inner nodes is 0 because of the requirements (i) and (ii).

The valuation of an upper outer node x is $v_1(\gamma)$ where γ is the only edge such that $\gamma(0) = x$. Similarly, the valuation of a lower outer node x is $v_1(\bar{\gamma})$ where γ is the only edge such that $\gamma(1) = x$. In both cases, we denote this valuation by $v_b(x)$. The **domain** of a valued graph is the tuple $(v_b(x_1), \ldots, v_b(x_p))$ where x_i is the i–th upper outer node of the graph. The **codomain** is defined similarly with the lower outer nodes.

We can define the replacement of the valued graphs G^1 by G^2 in the valued graph Γ, when the domains and codomains of G^1 and G^2 are the same and the valuations of the nodes of G^1 and Γ agree.

Lemma 1. Let Γ be a valued graph and G^1, G^2 be valued unit graphs. Suppose that t is an occurrence of G^1 in Γ, that the valuations of the nodes of Γ and $\phi_t(G^1)$ are identical and that the domains and codomains of G^1 and G^2 match. Then $\Gamma[G^1 := G^2]_t$ can be given a valuation, such that it agrees with the valuation of Γ on the nodes and edges included in $\Gamma - \phi_t(G^1)$, and it agrees with the valuation of $\phi_t(G^2)$ on the inner nodes and edges included in $\phi_t(G^2)$.

Proof. We admit that if $\gamma : [0,1] \to \Gamma$ is an edge and $t \in [0,1]$ is such that $\gamma(t)$ is not a singular point of $\gamma([0,1])$, then $\rho(\gamma) = \rho(\gamma_{|[0,t]}) + \rho(\gamma_{|[t,1]})$.

Up to a regular transformation, we can assume that the outer nodes of $\phi_t(G^1)$ are not singular nodes in Γ. Then, for every edge γ in Γ, there is a series of $t_0 < \cdots < t_k$ such that $t_0 = 0$, $t_k = 1$, and for $0 < i < k$, $\gamma(t_k)$ is an outer node of $\phi_t(G^1)$, and such that $\gamma((t_i, t_{i+1}))$ is either included in $\phi_t(G^1)$ or in $\Gamma - \phi_t(G^1)$.
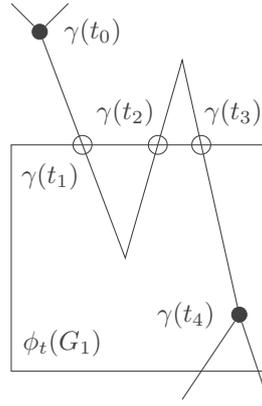
Moreover, identifying temporarily v_1 and v_b with their second projection, we have:

$$\rho(\gamma_{|[t_0,t_1]}) = v_b(\gamma(t_1)) - v_1(\gamma)$$

$$\text{for } 0 < i < k-1,\ \rho(\gamma_{|[t_i,t_{i+1}]}) = v_b(\gamma(t_{i+1})) - v_b(\gamma(t_i))$$

$$\rho(\gamma_{|[t_{k-1},t_k]}) = v_1(\bar\gamma) - v_b(\gamma(t_{k-1}))$$

Now for any edge γ in $\Gamma[G^1 := G^2]_t$, we have such a decomposition and $\rho(\gamma) = \sum_{i=0}^{k-1} \rho(\gamma_{|[t_i,t_{i+1}]})$. If $\gamma((0,t_1)) \subset \Gamma - \phi_t(G^2)$, we give γ the valuation of the edge in Γ starting on $\gamma(0)$ and whose image includes $\gamma((0,t_1))$. If $\gamma((0,t_1)) \subset \phi_t(G^2)$, then similarly we give γ the valuation from G^2. We can decompose the rotation number $\rho(\gamma)$ as follows:

$$\rho(\gamma) = \sum_{i=0}^{k-1} \rho(\gamma_{|[t_i,t_{i+1}]})$$

$$= v_b(\gamma(t_1)) - v_1(\gamma) + \sum_{i=1}^{k-2} \big(v_b(\gamma(t_{i+1})) - v_b(\gamma(t_i)) \big) + v_1(\bar\gamma) - v_b(\gamma(t_{k-1}))$$

$$= v_1(\bar\gamma) - v_1(\gamma)$$

Hence the condition (i) of the definition is satisfied. The two other conditions are directly inherited from Γ and G^2. $\qquad\square$

A.5 The category of valued graphs

Our autonomous category $L(\mathcal{C})$ will be defined as the category of $\mathcal{C}$-valued graphs. But to do so, we need to define an equivalence relation to account for some equalities of arrows from $\mathcal{C}$. This consists in defining reduction rules based on replacement.

We will need to define some particular $\mathcal{C}$-valued graphs. Instead of defining the graph and the valuations separately, we choose to draw them, replacing the black points representing our nodes by boxes containing the valuation of these nodes. The valuations of the edges are dropped when they are clear from the context.
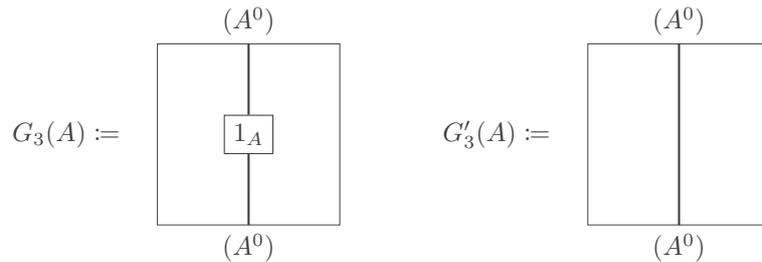
For all $f \in \mathcal{C}(A_1 \otimes \cdots \otimes A_n, B_1 \otimes \cdots \otimes B_p)$ and $g \in \mathcal{C}(B_1 \otimes \cdots \otimes B_p, C_1 \otimes \cdots \otimes C_q)$ we define the following graphs:
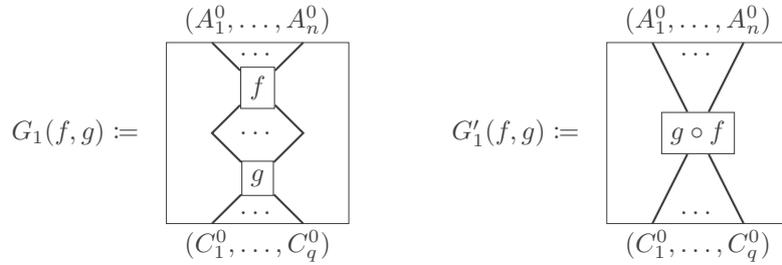
For all $f \in \mathcal{C}(A_1 \otimes \cdots \otimes A_p, B_1 \otimes \cdots \otimes B_q)$ and $g \in \mathcal{C}(C_1 \otimes \cdots \otimes C_n, D_1 \otimes \cdots \otimes D_m)$ we define the following graphs:
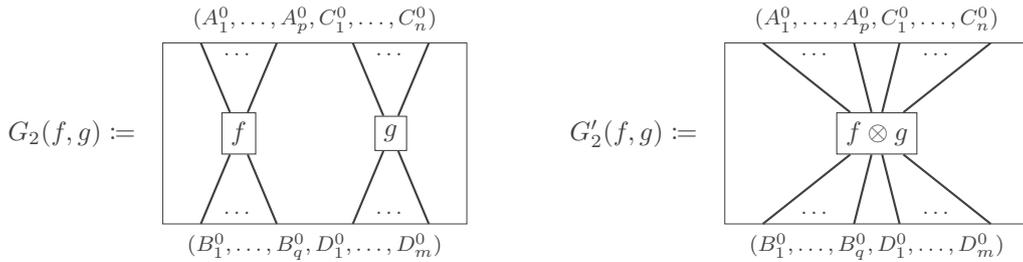
For $A \in \mathcal{C}$ we define the following graphs:

The generalized version of this last replacement pair (with multiple inputs and outputs) will be a consequence of the three replacement pairs, as it can be obtained as the n–fold product of identities.

Definition 6. Let A and B be two $\mathcal{C}$-valued yankable graphs. We say that A reduces to B (denoted by $A \leq B$) when A contains the sub-graph $G = G_i(f,g)$ for some f and g arrows of $\mathcal{C}$ (or $G = G_3(X)$ for some $X \in \mathrm{Ob}(\mathcal{C})$), and B can be obtained by replacing this occurrence of G by $G' = G'_i(f,g)$ (or $G'_3(X)$, respectively).

Another useful relation is $\Gamma \rightsquigarrow \Gamma'$, which holds when there is a deformation of graphs from Γ to Γ'. Finally we define the relation $\sim_{\mathcal{C}}$ as the reflexive, symmetric and transitive closure of $\leq \cup \rightsquigarrow$.

Definition 7. The category $L(\mathcal{C})$ has:

- objects of the form $(A_1^{n_1}, \ldots, A_p^{n_p})$ where $A_i \in \mathrm{Ob}(\mathcal{C})$ and $n_i \in \mathbb{Z}$

- morphisms $f : A \to B$, where f is an equivalence class under $\sim_{\mathcal{C}}$ of $\mathcal{C}$-valued unit graphs with inputs A and outputs B.

The composition and tensor product in $L(\mathcal{C})$ are defined as in Joyal and Street [1991]. Let $f \in L(\mathcal{C})(A, B)$ and $g \in L(\mathcal{C})(C, D)$ be $\mathcal{C}$–valued unit graphs. The morphism $f \otimes g \in L(\mathcal{C})(A \otimes_{L(\mathcal{C})} C, B \otimes_{L(\mathcal{C})} D)$ is defined by concatenating horizontally shrunken versions of f and g:

$$f \otimes g = \phi_{(0,\frac{1}{2},0,1)}(f) \cup \phi_{(\frac{1}{2},1,0,1)}(g).$$

Let $f \in L(\mathcal{C})(A, B)$ and $g \in L(\mathcal{C})(B, C)$ be $\mathcal{C}$–valued unit graphs, with $B = (B_1^{n_1}, \ldots, B_p^{n_p})$. We cannot simply stack the diagrams vertically to define the sequential composite $g \circ f$, because the horizontal positions $(u_1, \ldots, u_p)$ of the lower outer gates of f might not match with the positions $(v_1, \ldots, v_p)$ of the upper outer gates of g. Hence we add identity links between them:

$$f \circ g = \phi_{(0,1,0,\frac{1}{3})}(f) \cup \phi_{(0,1,\frac{1}{3},\frac{2}{3})}(L) \cup \phi_{(0,1,\frac{2}{3},1)}(g)$$

where L is the diagram with identity links between points $(u_i, 1)$ and $(v_i, 0)$ where u_i (respectively v_i) is the abscissa of the i–th lower outer gate of g (respectively upper outer gate of f).
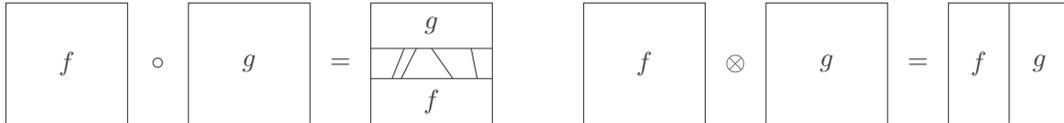
This category is well defined because the composition is compatible with the relation $\sim_{\mathcal{C}}$. In other words, the equivalence class under $\sim_{\mathcal{C}}$ of the vertical stacking of two graphs does not depend on the choice of the two representatives. As it is also the case for the horizontal concatenation, $L(\mathcal{C})$ is also strict monoidal.

The product on objects is the concatenation of lists, the unit object is the empty list denoted by (), and the tensor product on arrows is the horizontal juxtaposition. Formally, this product is different from the tensor product of $\mathcal{C}$. But the rewrite rules defined above provide a bridge between the two. If there are objects $A, B, C, D \in \mathcal{C}$ such that $A \otimes B = C \otimes D$, then the objects (A^0, B^0) and (C^0, D^0) are isomorphic in $L(\mathcal{C})$, with the following isomorphism:

$$\alpha_{A,B,C,D} = \qquad\qquad\qquad \alpha_{A,B,C,D}^{-1} = \alpha_{C,D,A,B}$$

Proposition 2. For any monoidal category $\mathcal{C}$, $L(\mathcal{C})$ is autonomous.

Proof. For any object $(A_1^{n_1}, \ldots, A_p^{n_p})$, let us show that it has a left adjoint $(A_p^{n_p-1}, \ldots, A_1^{n_p-1})$. A similar argument shows that it has a right adjoint $(A_p^{n_p+1}, \ldots, A_1^{n_1+1})$.

We define the following morphisms:

$$\epsilon : (A_p^{n_p-1}, \ldots, A_1^{n_1-1}, A_1^{n_1}, \ldots, A_p^{n_p}) \to () \qquad\qquad \eta : () \to (A_1^{n_1}, \ldots, A_p^{n_p}, A_p^{n_p-1}, \ldots, A_1^{n_1-1})$$

We emphasize that these graphs are valid $\mathcal{C}$-valued graphs and hence yankable: as they have no inner nodes, the condition of Definition 4 and the condition (ii) of Definition 5 are vacuously satisfied. They satisfy the yanking equalities (**??**), hence the category is autonomous. $\qquad\square$

B   Freeness of $L(\mathcal{C})$ over $\mathcal{C}$

What remains to do is to show that $L(\mathcal{C})$ is the free autonomous category generated by $\mathcal{C}$. The first step is to show that $\mathcal{C}$ can be embedded functorialy in $L(\mathcal{C})$. Then, assuming

that $\mathcal{C}$ is autonomous, we define the value of a $\mathcal{C}$-valued graph. Finally we show that these two constructions are adjoint, hence the freeness of $L(\mathcal{C})$.

B.1 Functorial embedding of the original category

We define a strongly monoidal functorial embedding of $\mathcal{C}$ in $L(\mathcal{C})$.

Definition 8. Let $F : \mathcal{C} \to L(\mathcal{C})$ be such that for all $A \in \mathrm{Ob}(\mathcal{C}), F(A) = (A^0)$ and

As a consequence of the first rewrite rule of our relation $\leq$, F is a functor. Moreover, it is strongly monoidal, with the natural isomorphism $\mu_{A,B} : (A^0, B^0) \to (A \otimes B^0)$ and the isomorphism $\lambda : () \to (I^0)$. The coherence equations translate into the following equalities, which hold from the rewrite rules defined earlier:

B.2 Value of a valued graph

Given a $\mathcal{C}$-valued graph, we cannot in general interpret this graph as an arrow of $\mathcal{C}$, because $\mathcal{C}$ is not always autonomous. But when $\mathcal{C}$ is autonomous, we can use the notion of **value** $v(\Gamma)$ of a valued graph Γ. We briefly recall its definition, taken from Joyal and Street [1988] and adapted to our terminology. We start with the definition of the value of a graph in the monoidal case. To do so, we restrict our graphs further by requiring that the edges are vertical, in the following sense.

Definition 9. A **progressive graph** is a graph Γ such that for all edge γ, the projection on the second coordinate of γ is injective.

As a consequence, the rotation numbers of edges are null in a progressive graph. To define the value of such a graph, we decompose it into simpler slices.

Definition 10. A **prime graph** is a progressive graph with exactly one inner node, and such that every edge in the graph is connected to it.

An **invertible graph** is a progressive graph with no inner node.

An **elementary slice** is a progressive graph Γ that can be decomposed as $G_1 \otimes \cdots \otimes G_n$ where for each $1 \leq i \leq n$, G_i is prime or invertible.
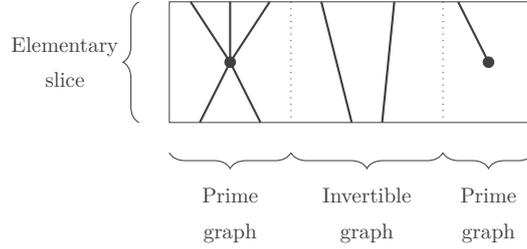

When these graphs are valued, we can give them a value $v(\Gamma)$. The value $v(\Gamma)$ of a prime graph Γ is the value of its unique inner node and the value of an invertible graph is the identity of its domain (which is equal to its codomain). Finally, we define the value of an elementary slice $\Gamma = G_1 \otimes \cdots \otimes G_n$ by $v(\Gamma) = v(G_1) \otimes \cdots \otimes v(G_n)$, which is independent of the decomposition.

Notice that any progressive graph Γ can be written as $\Gamma = G_1 \circ \cdots \circ G_p$ where G_i are elementary slices. To define the value of progressive graphs, we need the following lemma, whose proof is a direct consequence of Proposition 1.1 in Joyal and Street [1991].

Lemma 2. Let Γ be a progressive graph. If $\Gamma = G_1 \circ \cdots \circ G_p = G'_1 \circ \cdots \circ G'_q$ where the G_i and G'_j are elementary slices, then $v(G_1) \circ \cdots \circ v(G_p) = v(G'_1) \circ \cdots \circ v(G'_q)$

This defines the value of progressive graphs.

The value of general valued graphs is obtained by making the units and counits explicit in non progressive edges. More precisely, given a valued graph Γ and $\epsilon > 0$ we define the progressive graph Γ_ϵ by replacing each non progressive edge as follows. First, horizontal segments are eliminated, using the following replacements (and their upside-down counterparts):
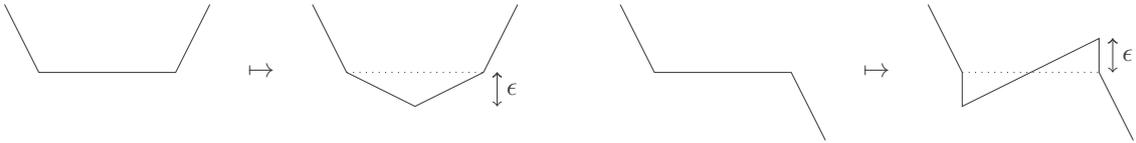
Then, the singular points at turns are replaced by inner nodes with the appropriate valuation. Let γ be an edge and be x a singular point in γ such that both of its adjacent segments are above x. The case where they are both below is similar. Let $a < b \in [0,1]$ be such that $[\gamma(a), x]$ and $[x, \gamma(b)]$ are strictly included in the adjacent segments of x. Finally let $(A, p) = v_1(\gamma)$ and $n = p + \rho(\gamma_{|[0,a]})$. As the category in which the graph is valued is autonomous, A has an $(n+1)$–fold right adjoint denoted by $A^{(n+1)}$, and there is an associated counit $\epsilon_{A^{(n+1)}} : A^{(n)} \otimes A^{(n+1)} \to I$. We replace x by an inner node valued by $\epsilon_{A^{(n+1)}}$. The case for the unit is symmetric.
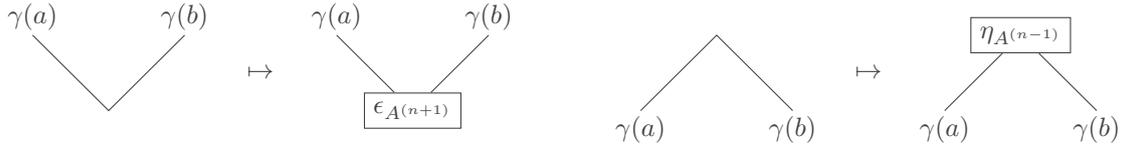
Note that these transformations are different from the notion of replacement intro-
duced earlier in Section A.3. Here, we only show how to convert a valued graph into
a progressive graph, but do not identify them using an equivalence relation as for the
replacement pairs of Section A.3.

It is shown in Joyal and Street [1988] that for ϵ small enough, $v(\Gamma_\epsilon)$ is independent
of ϵ. We define $\tilde{v}(\Gamma) \coloneqq v(\Gamma_\epsilon)$ for such an ϵ. This defines the value of $\mathcal{C}$–valued graphs
when $\mathcal{C}$ is autonomous. They also show in their Theorem 4 that this value is invariant
under deformations of graphs. We state the following lemma for later use:

Lemma 3. Let $f \in \mathrm{Mor}(L(\mathcal{C}))$ be a $\mathcal{C}$-valued graph, where $\mathcal{C}$ is autonomous. There is a $\mathcal{C}$-valued
progressive graph g such that $\tilde{v}(f) = \tilde{v}(g)$. Moreover, when f has no horizontal segment, g can be
obtained from f by replacing singular points by inner nodes.

In order to make $\tilde{v}$ a strict monoidal functor, we need to prove the invariance of $\tilde{v}$
under replacement.

Lemma 4. Let $f \in \mathrm{Mor}(L(\mathcal{C}))$ be a $\mathcal{C}$-valued graph, where $\mathcal{C}$ is autonomous. Let (G^1, G^2) be a
replacement pair such that $\tilde{v}(G^1) = \tilde{v}(G^2)$. Let t be an occurrence of G^1 in f. Then $\tilde{v}(f[G^1 \coloneqq G^2]_t) = \tilde{v}(f)$.

Proof. Let f be a valued unit graph, and $t = (a, b, c, d)$ be an occurrence of G^1 in f. As $\tilde{v}(f)$ is
invariant under deformation of f, we can assume that f contains no horizontal segment. Up to
another regular deformation, c and d are regular slices. Hence f can be decomposed into graphs
f_1, f_2 and f_3 with boundaries respectively $(0, c)$, (c, d) and $(d, 1)$. We have $f = f_1 \circ f_2 \circ f_3$ and
$f[G^1 \coloneqq G^2]_t = f_1 \circ f_2[G^1 \coloneqq G^2]_{t'} \circ f_3$, where $t' = (a, b, 0, 1)$. Similarly, f_2 can be decomposed
in the vertical slices g_1, g_2 and g_3, with vertical boundaries respectively $(0, a)$, (a, b), $(b, 1)$. We
have $f_2[G^1 \coloneqq G^2]_t = g_1 \otimes G^2 \otimes g_3$ hence $v(f_2[G^1 \coloneqq G^2]_t) = \tilde{v}(g_1) \otimes \tilde{v}(G^2) \otimes \tilde{v}(g_3)$ Finally, as
$\tilde{v}(G^1) = \tilde{v}(G^2)$ and $v(f) = \tilde{v}(f_1) \circ \tilde{v}(f_2) \circ \tilde{v}(f_3)$, we have the required invariance.   □

Hence, as $\tilde{v}$ is compatible with the two relations $\leq$ and $\rightsquigarrow$, it is compatible with $\sim_\mathcal{C}$. So
$\tilde{v} : L(\mathcal{C}) \to \mathcal{C}$ is defined and is a strict monoidal functor.

B.3   A pair adjoint functors

The objects introduced in our construction can be seen as part of an adjunction be-
tween a free and a forgetful functor. This will show that $L(\mathcal{C})$ has the required categor-
ical properties to be called the free autonomous category generated by the monoidal
category $\mathcal{C}$.

Let **Mon** be the category of strict monoidal categories and strong monoidal functors between them. We denote by **Nom** (as in auto*nom*ous) the category of autonomous categories and strong monoidal functors between them.[1] Our construction L corresponds to a functor from **Mon** to **Nom**. To make L a functor, we need to define how it translates a strong monoidal functor $f : \mathcal{C} \to \mathcal{D}$ to $L(f) : L\mathcal{C} \to L\mathcal{D}$. Let $\mu_{(A_1,\ldots,A_n)} : f(A_1 \otimes \cdots \otimes A_n) \simeq f(A_1) \otimes \cdots \otimes f(A_n)$ be coherence isomorphism associated to f (with $\mu_{()} : f(I_\mathcal{C}) \simeq I_\mathcal{D}$). We define $L(f)$ by

$$L(f) : (A_1^{n_1}, \ldots, A_p^{n_p}) \mapsto (f(A_1)^{n_1}, \ldots, f(A_p)^{n_p}))$$
$$(\Gamma, v_0, v_1) \mapsto (\Gamma, v_0', f \circ v_1)$$

We cannot simply define v_0' by $f \circ v_0$ as we have to compose with the coherence isomorphisms of f to ensure that the domain and codomain of the valuation match with the product of the valuation of the incoming and outgoing edges. Let x be a node and $(A_1, \ldots, A_p)$ (respectively $(B_1, \ldots, B_q)$) be the valuations of its input (respectively output) edges. We define

$$v_0'(x) = \mu_{(B_1,\ldots,B_q)} \circ (f \circ v_0(x)) \circ \mu_{(A_1,\ldots,A_p)}^{-1}$$

One can check that $(\Gamma, v_0', f \circ v_1)$ is indeed a $\mathcal{D}$-valued graph and that L is a strict monoidal functor.

We will show that this functor has a right adjoint $R : \mathbf{Nom} \to \mathbf{Mon}$, the inclusion functor. We define the unit for this adjunction by $\eta_\mathcal{C} \coloneqq F_\mathcal{C} : \mathcal{C} \to RL\mathcal{C}$ where F is the functorial embedding defined in Section B.1.

The counit $\epsilon_\mathcal{C} : LR\mathcal{C} \to \mathcal{C}$ corresponds to the value functor introduced in section B.2. It is defined on objects by $(A_1^{n_1}, \ldots, A_p^{n_p}) \mapsto A_1^{(n_1)} \otimes \cdots \otimes A_p^{(n_p)}$. In other words, the formal product is sent to the actual product of the original category, and the formal adjoints are sent to the actual adjoints. An arrow f, that is to say a $\mathcal{C}$-valued graph, is sent to its value $v(f)$.

We now move on to the proof of the unit–counit equations, starting with $(R\epsilon) \circ (\eta R) = 1_R$. First, $\eta_{R(\mathcal{D})}$ takes an arrow in an autonomous category, seen as a monoidal category, and bundles it in a diagram. Then, $R\epsilon_\mathcal{D}$ evaluates this diagram in $\mathcal{D}$ (which is possible because $\mathcal{D}$ is actually autonomous), and the result is seen as an arrow in $R\mathcal{D}$.

$$f \in \mathcal{C}(A, B) \quad \xmapsto{\eta_{R(\mathcal{D})}} \quad \boxed{\begin{matrix}(A^0)\\ \\ \fbox{f}\\ \\ (B^0)\end{matrix}} \quad \xmapsto{R(\epsilon_\mathcal{D})} \quad f$$

---

[1]Recall that these functors automatically preserve adjoints.

Hence the composition of the two is the identity. Let us show the remaining equality: $(\epsilon L) \circ (L\eta) = 1_L$.

$$(L\eta)_{\mathcal{C}} = L(\eta_{\mathcal{C}}) : L\mathcal{C} \to LRL\mathcal{C}$$
$$(A_1^{n_1}, \ldots, A_p^{n_p}) \mapsto ((A_1^0)^{n_1}, \ldots, (A_p^0)^{n_p})$$

The functor $L(\eta_{\mathcal{C}})$ applies $\eta_{\mathcal{C}}$ to the valuations of the graph, and the result is composed with the coherence morphisms so that the domains and codomains match with the incoming and outgoing edges. Graphically, this consists in adding inner boxes in each node, with morally the same inputs and outputs as the outer box it is contained in. Then, $\epsilon_{L(\mathcal{C})}$ evaluates the resulting graph in $L(\mathcal{C})$.

Theorem 3. The following equality holds: $(\epsilon L) \circ (L\eta) = 1_L$.

Proof. Let f be an arrow in $L\mathcal{C}$. Up to a deformation described in Section B.2, we can assume that it has no horizontal segment. As $RL\mathcal{C}$ is autonomous, we can apply Lemma 3 to $L\eta_{\mathcal{C}}(f) \in$ Mor$(LRL\mathcal{C})$: $\tilde{v}(L\eta_{\mathcal{C}}(f)) = \tilde{v}(g)$ where g is progressive and is obtained from $L\eta_{\mathcal{C}}(f)$ by replacing singular points by inner nodes. But $L\eta_{\mathcal{C}}(f)$ differs only from f by the valuations: the underlying graph is the same. Hence the decomposition of g into prime and invertible factors given by Lemma 2 induces a decomposition of f, where the factors are not necessarily prime or invertible however.

We prove that $\tilde{v}(g) = f$ by induction on the number of factors in the decomposition of g. If g is prime, let x be its unique inner node. There are two cases. If x is also an inner node in f, then g is indeed mapped to itself, as shown in Figure 3. Otherwise, x corresponds to a singular point in f and is labelled by a unit or a counit, and is mapped to itself as shown in Figure 4. If g is invertible, it is mapped to itself as well.

Now for the general case, suppose that $g = g_1 \circ g_2$ where g_1 and g_2 can be decomposed in a smaller number of factors (the case $g_1 \otimes g_2$ is analogous). As noted earlier, this induces a decomposition $L\eta_{\mathcal{C}}(f) = h_1 \circ h_2$ such that g_i is the progressive version of h_i. This induces in turn a decomposition $f = f_1 \circ f_2$ such that $h_i = L\eta_{\mathcal{C}}(f_i)$. By induction, $\tilde{v}(g_i) = f_i$. As $\tilde{v}$ is a strict monoidal functor, we get $\tilde{v}(g) = f$. $\qquad \square$

The notion of adjunction helps us to relate our construction with that of Preller and Lambek [2007] who describe the free autonomous category generated by a category. Let $L' : \mathbf{Cat} \to \mathbf{Mon}$ be the free monoidal category functor, and $R' : \mathbf{Mon} \to \mathbf{Cat}$ be the corresponding forgetful functor. By composition of the adjunctions, $L \circ L'$ is left adjoint to $R' \circ R$.
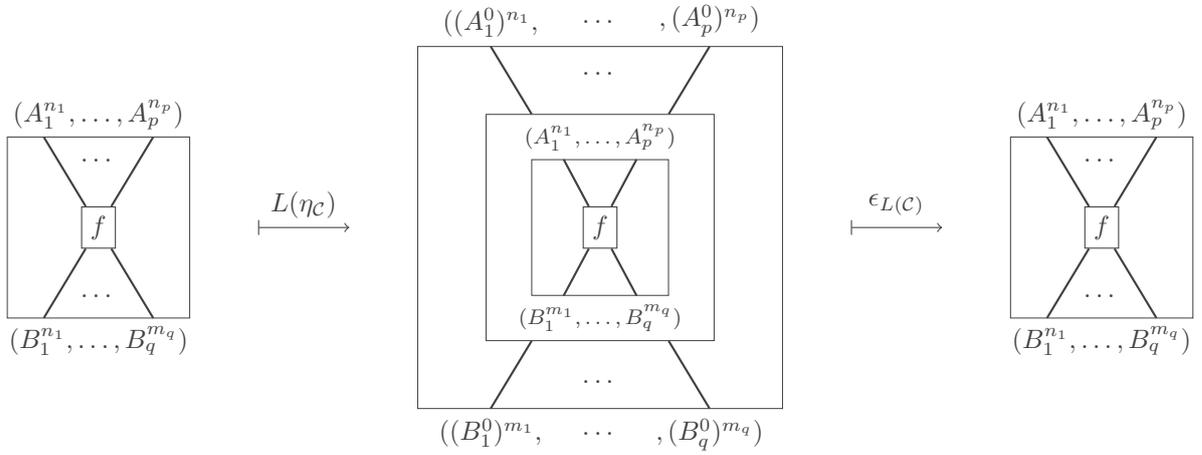
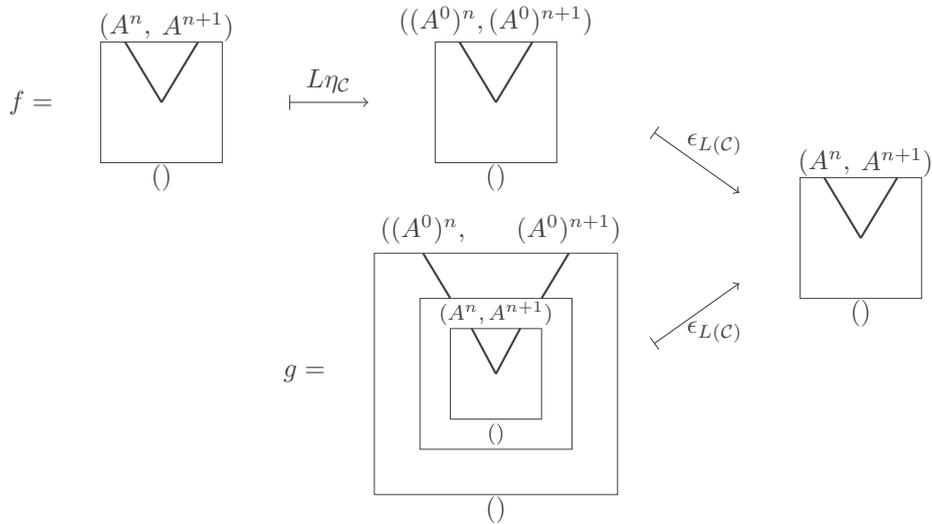Figure 3: Equality in the case of a prime diagram

Figure 4: Equality in the case of a counit

The construction of Preller and Lambek corresponds to a free functor from **Cat** to **Nom**, which is equivalent to $L \circ L'$ by uniqueness of the adjoint. Hence we just gave a factorization of their free functor.