String Diagrams for Layered Explanations

Leo Lobski Fabio Zanasi
University College London

We propose a categorical framework to reason about scientific *explanations*: descriptions of a phenomenon meant to translate it into simpler terms, or into a context that has been already understood. Our motivating examples come from systems biology, electrical circuit theory, and concurrency. We demonstrate how three explanatory models in these seemingly diverse areas can be all understood uniformly via a graphical calculus of *layered props*. Layered props allow for a compact visual presentation of the same phenomenon at different levels of precision, as well as the translation between these levels. Notably, our approach allows for *partial explanations*, that is, for translating just one part of a diagram while keeping the rest of the diagram untouched. Furthermore, our approach paves the way for formal reasoning about counterfactual models in systems biology.

1 Introduction

Different fields of science and engineering come with their own notions and traditions of explaining one phenomenon in terms of another one. For example, statistical mechanics explains thermodynamics, since it relies on fewer assumptions, which are moreover perceived as more fundamental than those of thermodynamics. A similar pattern may be found in the reduction of climate science to various areas of physics and biology. The converse move, from a "lower" to a "higher" level, is also interesting: for instance, temperature and vessel shape may be used to explain crystallisation. Choosing the right level of abstraction is paramount for successful communication between different disciplines, as well as between the scientific community and the general public. In particular, the definition of what constitutes an explanation is an increasingly important topic in the areas of automated reasoning and artificial intelligence [19].

Perhaps the most drastic divide between different modes of explaining can be found in biology, where some phenomena are explained *mechanistically* (or *reductively*), that is, by reducing them to the underlying chemical or physical laws, while others are explained *functionally*, that is, by appealing to what an organism does as a part of a larger whole [10, 21]. For instance, when explaining production of ATP within a cell, the mitochondria can either be introduced as elementary blocks providing energy to the cell (functional), or as compartments containing a whole pathway to process ATP (mechanistic). This divide is not merely of conceptual interest, but has practical implications for the modelling of biological systems: the ability to replicate biological functions is taken as a measure of success of the rule based models [8, 10]. However, the existing rule based languages that model molecular interactions are typically not able to formally distinguish between mechanistic and functional rules, as these exist at different levels of abstraction [10].

The goal of this work is to identify fundamental mathematical structures underlying explanations across different fields of science. Upon these structures, we develop a formalism that is able to describe the different levels of abstractions involved in an explanation, and account for more elaborate aspects such as the divide above. Additionally, we attempt to provide a uniform framework for *counterfactual reasoning* by allowing explanations that depend on what could potentially occur. Ability to model counterfactual dependencies is of interest in rule-based models of molecular interactions [13]. We shall

Submitted to:
ACT 2022

illustrate our approach by showing how it models case studies in diverse scientific areas.[1]

In our framework, explanations always concern a certain *process*. The process can be thought of as an actually occurring natural phenomenon, or a computation, or a rule in some formal system. An explanation should then consist of another process whose level of abstraction is strictly lower than that of the process being explained. In addition to the lower level process, an explanation should state in what way the two processes are related, for example by giving a translation from one to the other. Moreover, we want the explanations to be *modular* or *compositional*, in the sense that the same explanation may be reused multiple times in case different systems have equivalent subsystems, and that the explanations can be composed to create larger, more complicated explanations. The reason for requiring modularity is twofold. First, it allows explanations to be reused by potentially different areas, in much the same way lemmas and theorems in mathematics are used to develop different theories. Second, this allows for a certain efficiency, as we may be interested in explaining only a part of a large system; in such a case modularity allows us to focus on this one part only, instead of explaining the whole system.

The above requirements for what an explanation should be like lead naturally to *monoidal categories*, as these allow for both sequential and parallel composition of processes (i.e. morphisms in a category). We assume that the monoidal categories are partially ordered "by abstraction", so that more abstract theories (i.e. categories) are higher in the order. We want to be able to compose not just the processes but also the explanations, so that we require the categories and functors under consideration to have a monoidal structure. We thus arrive at a 2-category which is able to simultaneously talk about processes in all the individual categories (0-cells), translations between processes (1-cells), compositions of the processes and the translations, as well as rules or equations between the processes and the translations (2-cells). The definitions of an explanation (Definitions 7 and 8) use all of this structure. This is the motivation for what we call a *layered prop* (Definition 2).

It is worth noting that, in the categorical approaches inspired by the paradigm of functorial semantics, an explanation and what is being explained live in two separate categories, with some translation between them expressed as a functor — see e.g. [1, 4, 3]. Within this perspective, some equality or relation in the domain is explained by passing to the codomain (or vice versa). Our framework allows to treat such situations in a single language, staying within one category. The main technical advantage of our approach is that partial interpretations are built into our language from the very beginning, potentially reducing the amount of computation that is needed. More conceptually, unlike in the functorial semantics approach, working in our framework allows for counterfactual reasoning: since we can mix-and-match categories and morphisms, this gives the flexibility to ask such questions as *What would happen if p did not occur?*

Our contributions are organised as follows. In Section 2 we define layered props and outline their connection to the so-called internal string diagram construction. Section 3 briefly outlines how a layered prop can be interpreted in the bicategory of pointed profunctors. We give three definitions of an explanation in Section 4: one applies to 1-cells, another one to 2-cells, and the last one formalises counterfactual explanations. The remaining sections contain case studies formalised in our framework. Section 5 shows an example involving biology and chemistry. Section 6 shows the explanation of electrical circuit behaviour in terms of signal flow graphs — as it draws from the circuit theory developed in [3, 4], this example also clarifies how our approach relates to the 'functorial semantics' approaches. Finally, Section 7 presents a case study from concurrency, involving the explanation of CCS expressions.

[1] On the other hand, we do not delve into the philosophical ramifications of our approach. Rather, the aim is to offer an abstract formalisation of of existing intuitions, thus potentially providing precise tools for debating what a scientific explanation *should* be.

2 Layered Props

We shall build our language on *string diagrammatic* syntax: the standard representation of morphisms in (strict) monoidal categories [22]. Algebraic reasoning on string diagrams is typically formulated using props (**pro**duct and **p**ermutation categories), which are just symmetric strict monoidal categories with the natural numbers as objects — see e.g. [15, 12, 24] for an overview. In fact, in order to model the different layers involved in an explanation, we will need a more sophisticated concept: *layered props*.

Since we want to talk about "string diagrams in context", the context being a theory at a particular level of abstraction, we draw the string diagrams inside a rectangle which represents its context. This allows us to reason both *internally* with the string diagrams, as well as *externally* by pasting and piling the rectangles. In order to formalise such graphical intuition as a layered prop, we need the preliminary notions of a *system of sets* and a *layered monoidal theory*.

We begin with systems of sets, which we think of as contexts and translations between them. Fix a collection of sets Ω. An Ω-*type* is a finite list of pairs $(\omega_1, \alpha_1; \ldots; \omega_n, \alpha_n)$ where each ω_i is in Ω and each $\alpha_i \in \omega_i^*$ is an element in the free monoid on ω_i. Precisely, define Ω-types recursively as:

- the empty list ε is an Ω-type,
- if t is a type, $\omega \in \Omega$, and $\alpha \in \omega^*$, then $(t;\ \omega, \alpha)$ is an Ω-type.

We denote the collection of all Ω-types by $\mathtt{type}^{\Omega}$.

We call Ω a *system of sets* when it is equipped with a partial order, and, for each comparable pair $\omega \leq \omega'$, with a choice of a homomorphism $f : \omega'^* \to \omega^*$. Intuitively, as we think of the sets $\omega \in \Omega$ as contexts, the partial order is saying which contexts are more abstract, and the homomorphisms are translations from more abstract contexts to less abstract ones. We now introduce the counterpart of algebraic theories for monoidal categories (typically called monoidal theories, see e.g. [24]) based on this structure.

Definition 1 (Layered monoidal theory)**.** A *layered monoidal theory* is a tuple $(\Omega, \Sigma, \mathtt{ar}, \mathtt{coar})$ consisting of a system of sets Ω, a set Σ (*signature*), and functions $\mathtt{ar}, \mathtt{coar} : \Sigma \to \mathtt{type}^{\Omega}$.

It is convenient to introduce notation for the *internal signature* Σ^i, defined as

$$\Sigma^i \coloneqq \{\sigma \in \Sigma : \text{ there are } \omega \in \Omega \text{ and } \alpha, \beta \in \omega^* \text{ s.t. } \mathtt{ar}(\sigma) = (\omega, \alpha) \text{ and } \mathtt{coar}(\sigma) = (\omega, \beta)\}.$$

The idea is that the generators in Σ^i are completely contained in a single context ω: there is no transition between contexts involved. We define the *terms* and the corresponding *sorts* (arity-coarity pairs of types $(t \mid s)$) of a layered monoidal theory by the recursive procedure in Figure 1. For the $\otimes_\omega$-rule, there is a side condition that only the rules for Σ^i, identity, composition and $\otimes_\omega$ are used in constructing the terms x and y. This ensures that x and y only contain generators from the internal signature, so that it makes sense to graphically represent the term $x \otimes_\omega y$ as juxtaposition of x and y inside the rectangle representing ω. We call the terms that are generated using only these four rules *internal*. If a layered monoidal theory is generated by monoidal categories (see Section 2.1 below), the internal terms will correspond precisely to morphisms inside the categories.

We think of the *pants* and the *copants* (line 3 of Figure 1) as composition and decomposition within a level of abstraction. The black and white triangles (line 4 of Figure 1) are translations between the levels: ◀ translates an abstract layer to a more concrete one (*refinement*), while ▷ maps towards a higher abstraction (*coarsening*). In the pointed profunctor semantics (Section 3), pants will be interpreted as the monoidal product (seen as a profunctor), and copants as its adjoint profunctor (cf. axioms in Figure 3). Likewise, refinement will be interpreted as a monoidal functor (seen as a profunctor), and coarsening as its adjoint (cf. axioms in Figure 4).

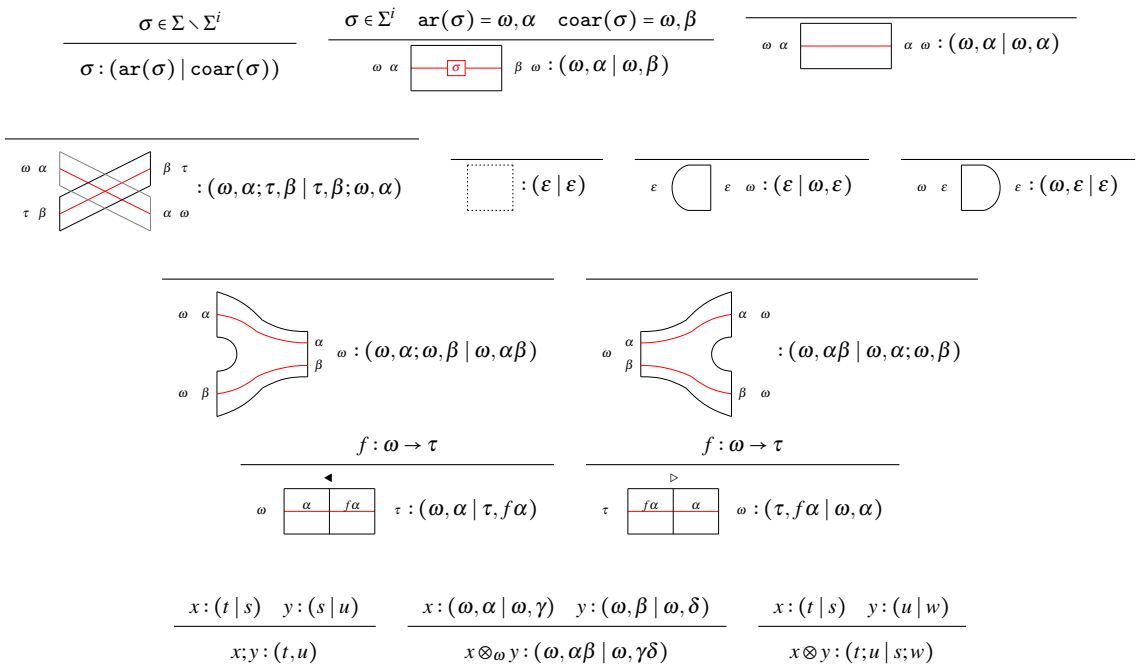
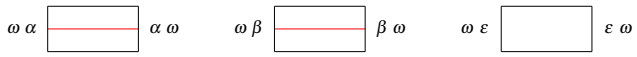


Figure 1: Recursive construction of the terms of a layered monoidal theory. Each term of the sort $(\omega, \alpha \mid \tau, \beta)$ is drawn as an area connecting the type ω, α on the left to the type τ, β on the right. The area inside a term, demarcated by black lines, is to be thought as representing the set ω, and an internal red wire as α (the element of ω^*). The change of type $\alpha \to \beta$ inside ω is drawn as a red box. The change of type at the level of sets $\omega \to \tau$ is drawn as a vertical black line.

In order to define a layered prop, we need to consider the terms modulo certain equations. Given $\omega \in \Omega$ and $\alpha, \beta \in \omega^*$, consider the internal terms with the sort $(\omega, \alpha \mid \omega, \beta)$. We may quotient this subset by the usual rules of monoidal categories: the identities and the monoidal unit are given by the third rule on the first line,



while the monoidal product $\otimes_\omega$ is represented by vertical juxtaposition inside the ω-rectangle. Further, we may quotient all the terms with the sort $(t \mid s)$ by the usual rules of symmetric monoidal categories: the identities are given by appropriate vertical juxtapositions of terms generated by the third rule on the first line, the monoidal unit is given by the second rule on the second line, and the monoidal product $\otimes$ is once again represented by vertical juxtaposition, this time of whole rectangles.

Definition 2 (Layered prop)**.** A *layered prop* generated by a layered monoidal theory $(\Omega, \Sigma, \mathtt{ar}, \mathtt{coar})$ is a 2-category whose 0-cells are the types $\mathtt{type}^\Omega$ and whose 1-cells $t \to s$ are terms with sort $(t \mid s)$ quotiented by the laws of symmetric monoidal categories both internally and externally, as discussed above. The 2-cells are generated by the rules in Figures 2, 3 and 4. Where arrows are going in both directions, we require the 2-cells to be inverses. Further, we require the usual triangle identities to hold for each unit-counit pair in Figure 3, and the usual laws of monoidal categories to hold for the isomorphisms in Figure 4.

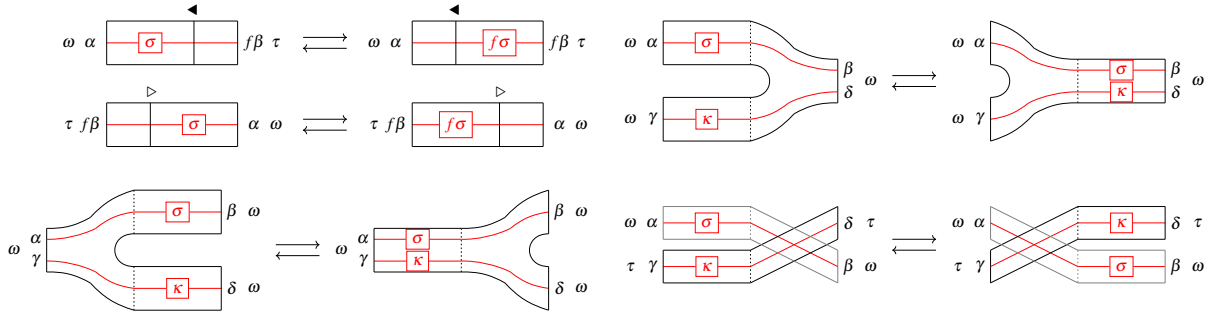


Figure 2: 2-cells of a layered prop expressing functoriality of refinement, coarsening, pants and copants.

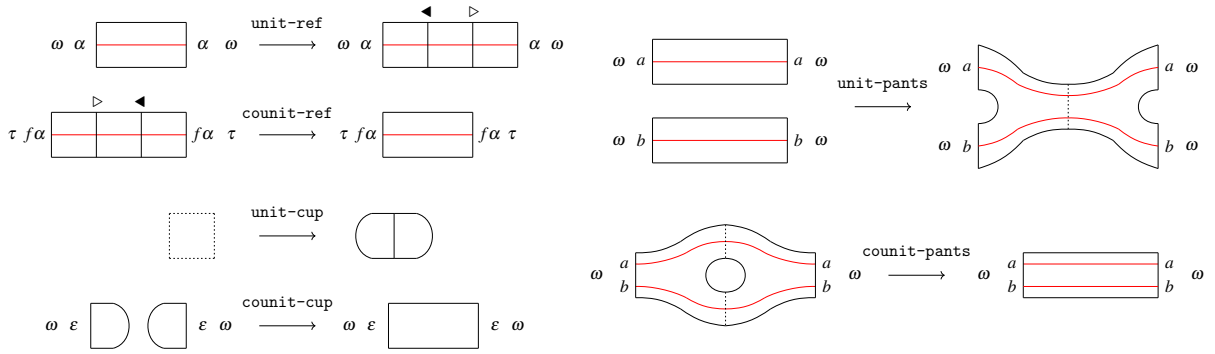


Figure 3: 2-cells of a layered prop that exhibit pants-copants and refinement-coarsening as two adjoint pairs.

Note that the 1-categorical structure of a layered prop can be seen as a generalisation of a coloured prop: any coloured prop gives rise to a layered prop with just one layer (i.e. with just one set in Ω). Furthermore, layered props are known in the literature as the *internal string diagram construction*. This was first introduced in the work of Bartlett, Douglas, Schommer-Pries and Vicary on topological quantum field theories [2]. The connection to profunctors is discussed by Hu [9].

2.1 Layered Props from Monoidal Categories

It is natural to build layered props from existing monoidal categories. In fact all the examples of layered props we consider arise in this way — see Sections 5-7 below. We assume that instead of a system of sets, we have a system of *monoidal categories* Ω with monoidal functors instead of homomorphisms. The construction of the layered prop $\mathcal{L}(\Omega)$ then proceeds as before, taking the internal signature to contain all morphisms in each category in Ω. We now proceed to define this formally.

A *system of monoidal categories* Ω is a subcategory of **Cat** such that

- every category $\omega \in \Omega$ is strict monoidal,
- every functor in Ω is strict monoidal,
- there is at most one functor between any pair of categories, that is, Ω is posetal.

The last condition is assumed merely for simplicity, we could construct a layered prop from Ω with multiple monoidal functors between a pair of monoidal categories. The formalism could be modified to incorporate non-strict monoidal categories, we leave this for future work.

We view a system of monoidal categories as a system of sets in a straightforward manner: the col-

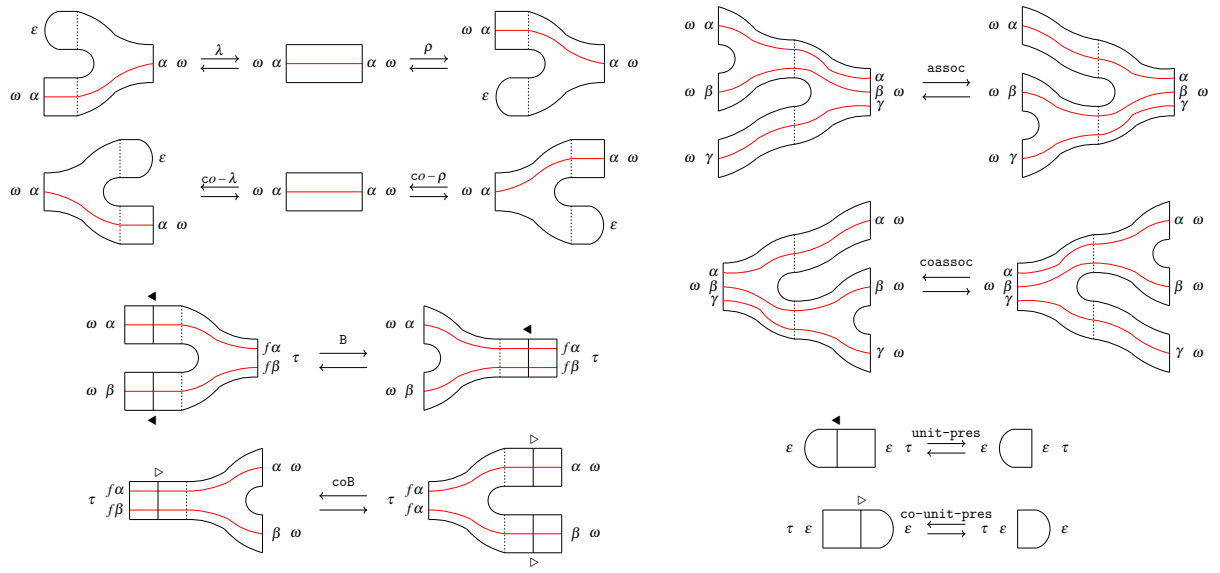


Figure 4: 2-cells of a layered prop that are motivated by monoidal categories and functors.

lection of sets is given by $\{\mathcal{O}b(\omega) : \omega \in \mathcal{O}b(\Omega)\}$, we identify $\alpha\beta \coloneqq \alpha \otimes \beta$ for all $\alpha, \beta \in \omega$ and $\omega \in \Omega$, we have $\omega \leq \omega'$ whenever there is a functor $f : \omega' \to \omega$, and the monoid homomorphisms are given by the restriction of each functor to objetcs.

Assuming that all the categories $\omega \in \Omega$ as well as Ω itself are small, we define the signature $\Sigma(\Omega)$ as follows:

$$\Sigma(\Omega) \coloneqq \left\{\sigma_\omega^{\alpha,\beta}\right\}_{\omega\in\mathcal{O}b(\Omega),\alpha,\beta\in\mathcal{O}b(\omega),\sigma:\alpha\to\beta}.$$

The arities and coarities are defined as:

$$\mathtt{ar}\left(\sigma_\omega^{\alpha,\beta}\right) = \omega, \alpha \qquad\qquad \mathtt{coar}\left(\sigma_\omega^{\alpha,\beta}\right) = \omega, \beta.$$

In other words, $\Sigma(\Omega)$ contains every morphism in every category $\omega \in \Omega$.

Definition 3 (Layered prop generated by a system of monoidal categories)**.** A *layered prop generated by a system of monoidal categories* Ω is the layered prop generated by the layered monoidal theory $(\Omega, \Sigma(\Omega), \mathtt{ar}, \mathtt{coar})$. Additional generators for 2-cells are given by the equalities of morphisms in each category $\omega \in \Omega$.

We denote the layered prop generated by a system of monoidal categories Ω by $\mathcal{L}(\Omega)$.

3 Pointed Profunctor Semantics

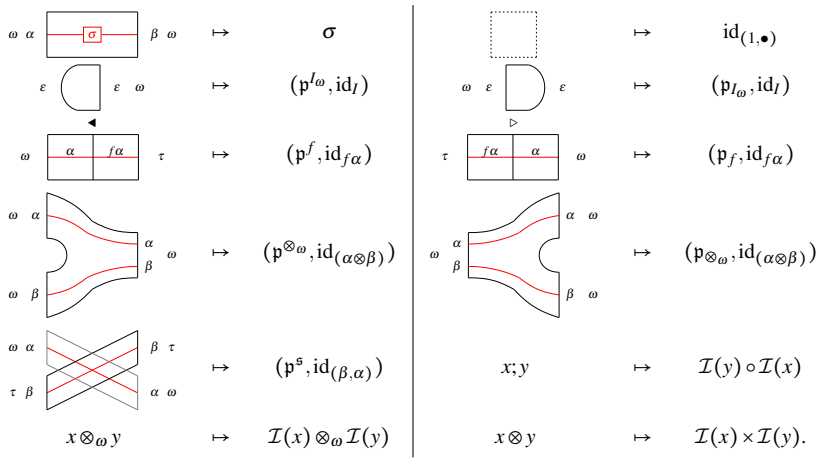
While our framework is purely syntactic (indeed, the whole point of constructing a layered prop is that we are able to treat all the layers in the same language), we are able to provide a semantic justification for the layered prop formalism: as we show in this section, they can be naturally interpreted in the category of pointed profunctors $\mathbf{Prof}_*$. We include the Appendix A on profunctors and pointed profunctors as a quick reference and to disambiguate any notation. For a proper introduction, see Borceux [6] and Loregian [14], and references therein.

Definition 4. Let $\mathcal{L}$ be a layered prop. A *profunctor model* of $\mathcal{L}$ is a 2-functor $\mathcal{L} \to \mathbf{Prof}_*$ which is consistent in the sense that

- if the 0-cells (ω, α) and (ω, β) are respectively mapped to $(\mathcal{C}, c)$ and $(\mathcal{D}, d)$, then $\mathcal{C} = \mathcal{D}$,
- if the 1-cells $\omega\ \alpha$ $\boxed{\sigma}$ $\beta\ \omega$ and $\omega\ \alpha$ $\boxed{\sigma'}$ $\beta\ \omega$ are respectively mapped to (P, f) and (Q, g), then $P = Q$.

For the rest of this section, we assume that Ω is a system of monoidal categories. We will show that there is a natural profunctor model of the layered prop generated by Ω. To this end, we wish to define a 2-functor $\mathcal{I} : \mathcal{L}(\Omega) \to \mathbf{Prof}_*$.

Let us define $\mathcal{I}$ on objects (i.e. Ω-types) recursively as follows: $\mathcal{I}(\varepsilon) \coloneqq (1, \bullet)$, and $\mathcal{I}(t; \omega, \alpha) \coloneqq \mathcal{I}(t) \times (\omega, \alpha)$. In order to define $\mathcal{I}$ on morphisms, for each $\omega \in \Omega$, let us write $I_\omega : 1 \to \omega$ for the functor sending the unique object of 1 to the monoidal unit of ω. Likewise, let us write $\mathfrak{s} : \mathcal{C} \times \mathcal{D} \to \mathcal{D} \times \mathcal{C}$ for the symmetry map in **Cat**. Note that since $\mathfrak{s}$ is an isomorphism, we have $\mathfrak{p}_\mathfrak{s} \simeq \mathfrak{p}^\mathfrak{s}$. We then define $\mathcal{I}$ by the following action on the generators:



where $\mathfrak{p}^-$ and $\mathfrak{p}_-$ are the covariant and contravariant embeddings of **Cat** in the category of profunctors, and σ stands for the pointed profunctor $(\hom_\omega, \sigma)$. We prove the following proposition in Appendix B.

Proposition 5. *The assignment $\mathcal{I}$ is a profunctor model of $\mathcal{L}(\Omega)$. Namely, it preserves the equalities of morphisms in each category $\omega \in \Omega$ as well as the rules in Figures 2, 3 and 4.*

4 Explanations

Using the formalism introduced in the previous sections, we are now able to formulate precisely the notion of an explanation. First, we give names to two special shapes of 1-cells in a layered prop and outline their connection to explanations. We assume that we are working with a layered prop generated by a system of monoidal categories Ω.

Definition 6 (Window, cowindow)**.** A *window* is a morphism in a layered prop of the form on the left below. Dually, a *cowindow* is a morphism in a layered prop of the form on the right below.

Windows correspond to *reductive* explanations: a process at the higher level gets translated to the lower level, where we can apply laws or rules that are (presumably) more flexible, after which we translate back to the higher level, hence completing the explanation. This remark should be compared to the shape of the explanation of glucose phosphorylation in Section 5 below.

Cowindows, in turn, correspond to *functional* explanations: a process at the lower level is justified by passing through a higher level in such a way that the higher level process translates back to what is being explained. It can thus be thought that the lower level process takes place in order to yield the appropriate form at the higher level. Axioms `unit-ref` and `counit-ref` state that there is an asymmetry between reductive and functional explanations: using `unit-ref`, it is always possible to create a window (and hence give a reductive explanation), while `counit-ref` only allows reducing a trivial cowindow to the identity. Note that what we call a cowindow is usually called a *functorial box* in the literature — see e.g. [16].

We may now define what an explanation is: we do this separately for 1-cells and for 2-cells. Both correspond to a *reduction*: an explanation of a 1-cell reduces a process to another one at a lower level of abstraction, while an explanation of a 2-cell reduces a rule between two processes to a rule between reductions of these processes. For examples of explanations (now in a formal sense), see Figure 6 and the discussion in Section 7 (1-cell), and Figure 7 (2-cell).

Definition 7 (Explanation of a 1-cell)**.** Let $\mathfrak{e}$ and σ be parallel 1-cells in a layered prop (that is, having the same domain and codomain). We say that $\mathfrak{e}$ is an *explanation* of σ if

1. σ is an internal morphism contained in some category $\omega \in \Omega$,
2. every internal non-identity morphism of $\mathfrak{e}$ is contained in some category ω' such that $\omega' < \omega$ in the partial order of Ω,
3. there is either a 2-cell $\mathfrak{e} \to \sigma$ or a 2-cell $\sigma \to \mathfrak{e}$.

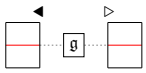
Definition 8 (Explanation of a 2-cell)**.** Let η and μ be parallel 2-cells in a layered prop. We say that η is an *explanation* of μ if

1. μ is generated by an equality of morphisms in some category $\omega \in \Omega$,
2. η can be constructed using the generating 2-cells of a layered prop and the 2-cells that come from an equality of morphisms in those categories ω' for which $\omega' < \omega$ in the partial order of Ω.

The above definitions correspond to the intuitive understanding of a (reductive) explanation we outlined in Section 1. The first condition in both definitions ensures that what is being explained is internal to a particular category, that is, to a description at a particular level of abstraction. The second condition says that the explanation is indeed reductive: it may only use lower levels of description than what is being explained (in addition to the metalanguage of the layered prop). This implies that an explanation must contain at least one window. The third condition in the first definition ensures that the explanation is *relevant* in the sense that it is either a sufficient or a necessary cause for what is being explained. There is no such condition for the explanations of 2-cells since in our setup there are no 3-cells. We thus simply assume that an explanation is relevant. This assumption need not be made if we are working with higher categories. These definitions can be dualised, this gives definitions of functional explanations, or "coexplanations". We will not need these in this work, and therefore omit the explicit statements.

Interestingly, if we require that the third condition of Definition 7 does not hold (i.e. there is no 2-cell between the explanation and what is being explained), we obtain the definition of a *counterfactual explanation*. Ability to model counterfactual reasoning is important for the causal analysis in the rule-based models of molecular interactions, such as the Kappa language [13]. While a particular simulation of a rule-based model may tell us that a rule $\mathfrak{e}$ was invoked in the computation of the effect σ, so that $\mathfrak{e}$ explains σ in the sense of Definition 7, this tells nothing about necessity (or sufficiency) of $\mathfrak{e}$ for σ. Thus

a rule-based model is not (without modifications) able to deal with such questions as *Would* σ *occur had* $\mathfrak{e}$ *not occurred?* In a layered prop, the positive answer to such question (establishing non-necessity) can be provided by finding a counterfactual explanation of σ that has the same sort as $\mathfrak{e}$. Intuitively, a (possibly) counterfactual explanation can be thought of as a 1-cell that "fills in the gap" left by $\mathfrak{e}$:



We give an example of a counterfactual explanation in our discussion of concurrency in Section 7.

Models based on variable substitution [18] and trajectory sampling [13] have been proposed to model counterfactual statements. Since our setup remains agnostic about what the internal morphisms in a layered prop actually are, we expect that both of these situations can be modelled within a layered prop. We leave this investigation for future work.

5 Example: Glucose Phosphorylation

In this section, we construct a minimal example — inspired by Krivine [11] — that illustrates our notion of an explanation (specifically, Definition 7) for an important biochemical process known as *phosphorylation of glucose*. This is motivated by the problem of systematising a vast amount of experimental data in systems biology in a way that is easy for humans to both understand and use. Our strategy is to define three monoidal categories that are capable of talking about chemical reactions at three different abstraction levels:

$\mathcal{L}^+$	English names of the relevant molecules
$\mathcal{M}ol^+$	Molecules
$\mathcal{P}art\mathcal{M}ol^+$	Partitions of molecules into smaller units

First, let us define $\mathcal{L}^+$ as the free monoidal category with generating objects

$$\{glucose, ATP, glucose\text{-}6\text{-}phosphate, ADP, hydrogen\ ion\},$$

whose monoidal product is denoted by $+$, and with just one generating morphism

$$glucose + ATP \longrightarrow glucose\text{-}6\text{-}phosphate + ADP + hydrogen\ ion. \tag{9}$$

The generating morphism simply represents the high-level chemical rule describing phosphorylation of glucose. Here *ATP* and *ADP* stand for *adenosine triphosphate* and *adenosine diphosphate*.

5.1 Molecules and Molecule Partitions

We define a *molecule partition* as a certain connected multigraph (Definition 10). We then identify as *molecules* those molecule partitions that do not have free variables. Fix a countable set of *free variables* FW. We denote the elements of FW by lowercase Greek letters $\alpha, \beta, \gamma, \ldots$. Let us define the set of *atoms* as containing the symbol for each main-group element of the periodic table together with the symbols $-$ and $+$: $\mathrm{At} \coloneqq \{-, +, H, C, O, P, \ldots\}$. Define the function $\mathbf{v} : \mathrm{At} \sqcup \mathrm{FW} \to \mathbb{N}$ as taking each element symbol to the valence of that element[2], define $\mathbf{v}(-) = \mathbf{v}(+) = 1$ and finally for all $\alpha \in \mathrm{FW}$ let $\mathbf{v}(\alpha) = 1$.

[2]This is a bit of a naive model, as valence is in general context-sensitive and not determined by a single atom. Yet this is good enough for the purposes of this example.

Definition 10 (Molecule partition)**.** A *molecule partition* is a triple (V,τ,m), where V is a finite set of *vertices*, $\tau : V \to \mathrm{At} \sqcup \mathrm{FW}$ is a function taking each vertex to its *type* and $m : V \times V \to \mathbb{N}$ is a function satisfying the following conditions:

- for all $v \in V$, we have $m(v,v) = 0$,
- for all $v,w \in V$, we have $m(v,w) = m(w,v)$,
- for all $v,u \in V$ with $v \neq u$, there are $w_0,\ldots,w_n \in V$ such that $w_0 = v$ and $w_n = u$ and $m(w_{i-1},w_i) \neq 0$ for each $i = 1,\ldots,n$,
- for all $v \in V$, we have $\sum_{u \in V} m(u,v) = \mathbf{v}\tau(v)$.

In other words, the integers $m(i,j)$ form an adjacency matrix of an irreflexive, symmetric and connected multigraph, and the sum of each row or column gives the valence of the (type of) corresponding vertex.

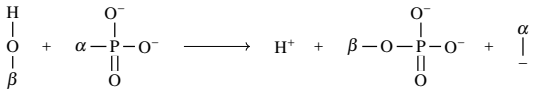
Definition 11 (Molecule)**.** We say that a molecule partition (V,τ,m) is a *molecule* if the image of the function τ is contained in At.

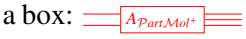
We denote the set of all molecules by $\mathcal{M}ol$ and the set of all molecule partitions by $\mathcal{P}art\mathcal{M}ol$. Define the *partitioning relation* $R \subseteq \mathcal{P}art\mathcal{M}ol \times (\mathcal{P}art\mathcal{M}ol \times \mathcal{P}art\mathcal{M}ol)$ as follows. Let $M = (V,\tau,m)$ be a molecule partition, let $u,v \in V$ and let $\alpha \in \mathrm{FW}$. Denote by $m' : V \times V \to \mathbb{N}$ the function such that $m'(u,v) = m'(v,u) = 0$ and $m' = m$ otherwise. Suppose that the following conditions are satisfied:

1. $m(u,v) = 1$,
2. the graph (V,m') is not connected,
3. α does not appear as a free variable in M (that is, $\tau(w) \neq \alpha$ for all $w \in V$).

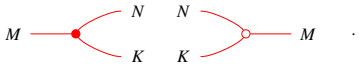
In such case we denote by $V(u)$ and $V(v)$ the connected components of u and v, respectively, in (V,m'). Let $M_u^\alpha = (V(u) \sqcup \{\alpha\}, \tau_\alpha, m_u)$ be the molecule partition where $\tau_\alpha(\alpha) = \alpha$ and $\tau_\alpha = \tau$ otherwise, and $m_u(u,\alpha) = m_u(\alpha,u) = 1$ and $m_u = m$ otherwise. The molecule partition $M_v^\alpha = (V(v) \sqcup \{\alpha\}, \tau_\alpha, m_v)$ is defined similarly. Now we finally define R by stipulating that $MR(M_u^\alpha, M_v^\alpha)$ for all M, v, u and α that satisfy the above conditions.

Let us define $\mathcal{M}ol^+$ as the free monoidal category with generating objects $\mathcal{M}ol$ and just one generating morphism, which has the same shape as the generating morphism of $\mathcal{L}^+$ (9), except that all the English names of the molecules are translated to the corresponding graphs (see Figure 5). Similarly, define $\mathcal{P}art\mathcal{M}ol^+$ as the free monoidal category with generating objects $\mathcal{P}art\mathcal{M}ol$. For all variables α and β we add the rule



as a generating morphism to $\mathcal{P}art\mathcal{M}ol^+$. We draw this as a box: 

. Further, for all molecule partitions M,N and K such that $MR(N,K)$ we introduce the following generators



We now wish to define monoidal functors $\mathcal{L}^+ \xrightarrow{T} \mathcal{M}ol^+ \xhookrightarrow{i} \mathcal{P}art\mathcal{M}ol^+$ so as to make this into a system of monoidal categories. First, define a monoidal functor $T : \mathcal{L}^+ \to \mathcal{M}ol^+$ by the action on the generating objects in Figure 5, where we use the convention from chemistry that an unlabelled vertex represents a carbon atom with an appropriate number of hydrogen atoms attached to it to make its valence equal to 4. The only generating morphism of $\mathcal{L}^+$ is mapped to the only generating morphism of $\mathcal{M}ol^+$. The monoidal functor $\mathcal{M}ol^+ \xhookrightarrow{i} \mathcal{P}art\mathcal{M}ol^+$ is identity on objects and maps the only generating morphism of $\mathcal{M}ol^+$ to the composite morphism in the middle rectangle of Figure 6.

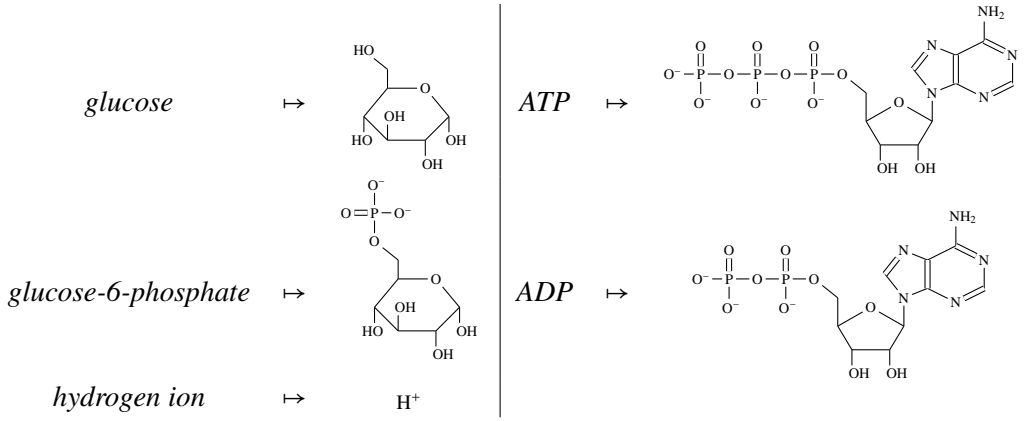


Figure 5: Translation of English names to chemical graphs.

5.2 Explaining Phosphorylation

We can now use the lowest level language $\mathcal{P}art\mathcal{M}ol^+$ to explain the high-level rule (9) as is shown in Figure 6. Note that this is indeed an explanation according to Definition 7, since the rule that is being explained is internal to $\mathcal{L}^+$, the explanation does not use any non-identity morphisms from $\mathcal{L}^+$, and the explanation can be derived starting from the rule (9) using the 2-cells of the layered prop, whose composite gives a 2-cell from the rule to the explanation. While the diagram in Figure 6 fulfills the

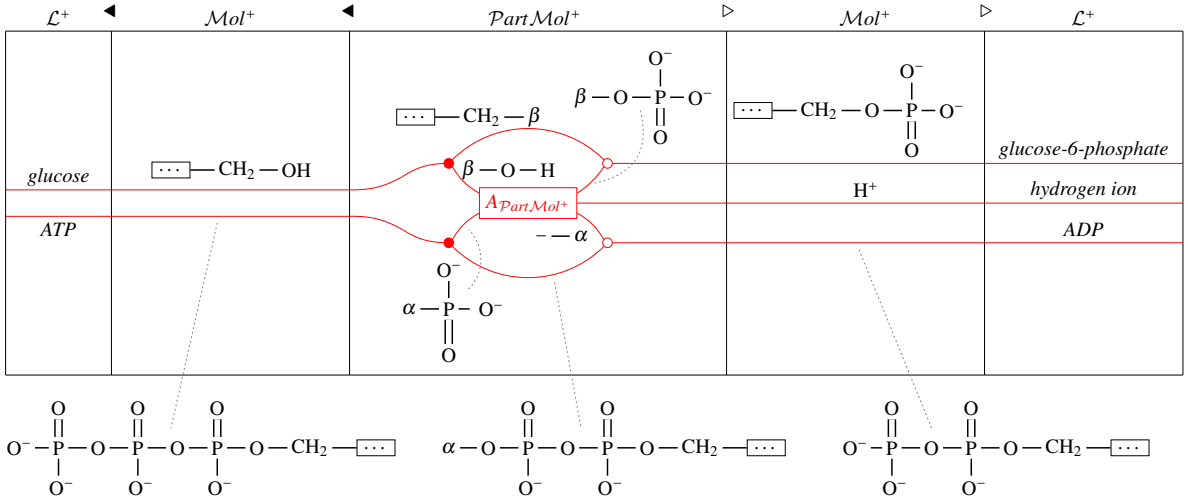


Figure 6: Explaining glucose phosphorylation: each area between the vertical black bars represents a layer, so in this case $\mathcal{L}^+$, $\mathcal{M}ol^+$ or $\mathcal{P}art\mathcal{M}ol^+$.

definition of an explanation, it is not very "explanatory" in an intuitive sense. This is because we chose to stop at a fairly high level of abstraction. It is important to note that the morphism $A_{\mathcal{P}art\mathcal{M}ol^+}$ is just a black box, which could itself be explained at the level of atoms exchanging electrons. Modularity of layered props would then allow us to add this further level to the diagram. The resulting explanation would bring us closer to satisfactorily answering the question *Why does this reaction occur?*.

We conclude this example by remarking that we didn't have to assume that we already know the higher level chemical rule (9). Instead we could have chosen to *generate* the higher level rules by

declaring as morphisms every 1-cell from $(\mathcal{L}^+, c)$ to $(\mathcal{L}^+, d)$ for some objects $c, d \in \mathcal{L}^+$. Instead of an explanation, this would correspond to deriving higher level rules from a single lower level rule.

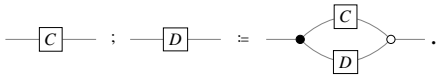
6 Example: Electrical Circuits

While in the previous section we constructed a minimal example from scratch, in this section we take an existing example from the literature where explanations are already used implicitly. Namely, we focus on the research program that has formalised electrical circuits in terms of string diagrams and given them an interpretation in graphical affine algebra [1, 4, 5, 3].

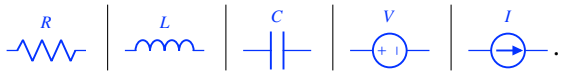
The string diagrammatic electrical circuit theory is a paradigmatic example of explanations taking a functorial form: the relations between electrical components are proved by interpreting them as morphisms in the graphical affine algebra. Thus this example also shows how functorial explanations can be incorporated into our framework. Note, however, that Boisseau and Sobociński [3] already use something like layered explanations to only partially translate their diagrams. They call the notational device used for this an *impedance box*. In our language, impedance boxes arise in a principled way as instances of a general definition: they are just windows (Definition 6) of a particular shape.

We define the props *graphical affine algebra* **GAA** and *electrical circuits* **ECirc** as well as the translation functor $\mathcal{I} : \mathbf{ECirc} \to \mathbf{GAA}$ as in [3], except that we quotient the morphisms in **ECirc** by equality under $\mathcal{I}$. This makes $\mathcal{I}$ faithful, which we reflect in our syntax by adding a left inverse to the 2-cell `unit-ref` in Figure 3[3]. Additionally, we define the *impedance category* **Imp** and the category of bipoles **Bip** in order to express the impedance calculus of [3] formally within our setup.

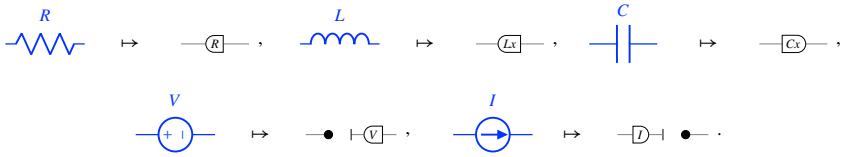
Definition 12 (Impedance category)**.** The *impedance category* **Imp** is a prop whose generating morphisms are all the morphisms of **GAA** with exactly one input and exactly one output. The identity is —• ○—, and composition is given by the rule



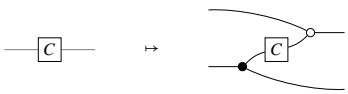
Definition 13 (Bipole category)**.** The *bipole category* **Bip** is the subcategory of **ECirc** given by those generators which have exactly one input and one output. That is, it is the free prop generated by

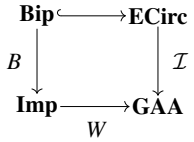


Define the "boxing" functor $B : \mathbf{Bip} \to \mathbf{Imp}$ by the following action on the generators:



Further, define a "wrapping" functor $W : \mathbf{Imp} \to \mathbf{GAA}$ which acts as $n \mapsto 2n$ on objects and on morphisms as shown below left. The boxing and the wrapping functors are so defined that we have a commutative square below right:

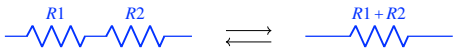




[3]This causes some problems for the semantic interpretation of Section 3, whose resolution we leave for a more technical paper.

where the top horizontal morphism is the inclusion functor, and $\mathcal{I}$ is the translation of electrical circuits to graphical affine algebra. Treating the above diagram of monoidal functors as a system of monoidal categories, we obtain a layered prop. Within this layered prop, we are able to replicate what is called the *impedance calculus* in [3]. To illustrate this, we give an explanation of the rule governing the sequential composition of resistors. This rule is a 2-cell in the layered prop, and the explanation is therefore that of a 2-cell (Definition 8).

Figure 7 shows how the rule for composing two resistors



can be explained (this is essentially part (i) of Proposition 3 of [3]). This is indeed an explanation of a 2-cell (Definition 8), since we are explaining an equality in **ECirc** using only the 2-cells of a layered prop and a 2-cell from **Imp** (the third 2-cell of the derivation).

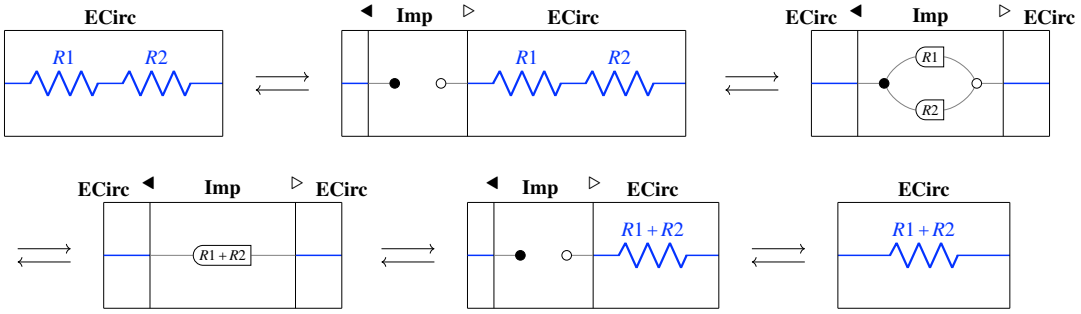


Figure 7: Explaining sequential composition of resistors. Note that the explanation relies on the composition in **Imp**. This could, in turn, be itself explained by translating to **GAA**.

As for the example with glucose phosphorylation, we could choose to generate the equalities in **ECirc** rather than assume them a priori. In this case, there would be no need to quotient morphisms in **ECirc** by equality under the translation functor, yet the equality of 1-cells should be taken up to a trivial window.

7 Example: Calculus of Communicating Systems

The calculus of communicating systems (CCS) [17] is widely used to reason about programs, formal languages and concurrency. Here we consider a restricted version of CCS and two ways to give semantics to the CCS expressions: *reduction semantics* is very heavily syntactic, in addition to the structural congruences, it only allows for only one rewrite rule (the *reduction*), while the *labelled transition system* (LTS) semantics [17] is more flexible and comes with more rewrite rules. Our goal is to show how the LTS semantics can be used to give an explanation (this time in the sense of Definition 7) of the rewrite rule of the reduction semantics. Intuitively, the LTS semantics may be seen as a lower level implementation of the concurrent processes described abstractly by CCS. Furthermore, we demonstrate that LTS semantics has a larger scope of allowed derivations than the reduction semantics by giving a counterfactual explanation of a rewrite rule in the reduction semantics.

Let us fix a set of *action names* A. Define $\bar{A} \coloneqq \{\bar{a} : a \in A\}$ and $Act \coloneqq A \cup \bar{A} \cup \{\tau\}$. The set of *processes* is defined recursively as follows, where x ranges over Act:

$$P ::= \quad 0 \;\Big|\; x.P \;\Big|\; P \parallel P.$$

Definition 14 (Congruence)**.** Define the *congruence* as the smallest equivalence relation $\sim$ on the set of processes that satisfies:

$$P \parallel Q \sim Q \parallel P, \qquad (P \parallel Q) \parallel R \sim P \parallel (Q \parallel R),$$

$$0 \parallel P \sim P, \qquad \text{if } P \sim P' \text{ and } Q \sim Q', \text{ then } P \parallel Q \sim P' \parallel Q'.$$

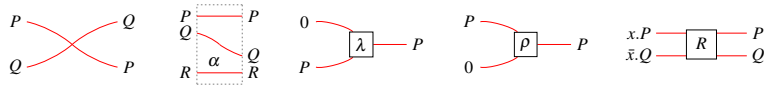
Definition 15 (Reduction semantics)**.** A *rewrite rule* in *reduction semantics* is an ordered pair of processes, which we write as $P \to Q$, generated by the following three *deduction rules*:

$$\frac{}{x.P \parallel \bar{x}.Q \to P \parallel Q} \qquad \frac{P \to Q}{P \parallel R \to Q \parallel R} \qquad \frac{P \to Q \quad P \sim P' \quad Q \sim Q'}{P' \to Q'}$$

In other words, rewrite rules are parallel compositions of the *reduction* (first rule in the above definition) up to the congruence. For instance, we can derive the following rewrite rule:

$$x.0 \parallel (y.0 \parallel \bar{x}.0) \to 0 \parallel (y.0 \parallel 0). \tag{16}$$

In order to talk about layered props, we wish to express reduction semantics as a monoidal category. Let **Red** be the monoidal category whose objects are the processes, monoidal product on objects is the parallel composition $\parallel$, and whose morphisms are generated by:



together with inverses for the first four generators. Here P, Q and R range over processes, and x ranges over A. The first four morphisms correspond to the congruence, and R corresponds to the first deduction rule for transitions. The parallel composition is taken care of by the monoidal structure. Note that the monoidal product is not strictly associative, so we need to keep track of the bracketing of the wires.

Next, we introduce a LTS as an alternative semantics for the above fragment of CCS.

Definition 17 (Labelled transition)**.** A *labelled transition* is a triple (P, x, Q), where P and Q are processes and $x \in Act$, generated by the deduction rules below. We write $P \xrightarrow{x} Q$ for such triple. Note that we write the silent action τ as an unlabelled arrow.

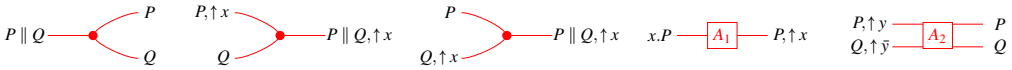
$$\frac{P' \xrightarrow{x} P}{P' \parallel Q \xrightarrow{x} P \parallel Q} \qquad \frac{P' \xrightarrow{x} P}{Q \parallel P' \xrightarrow{x} Q \parallel P} \qquad \frac{}{x.P \xrightarrow{x} P} \qquad \frac{P' \xrightarrow{x} P \quad Q' \xrightarrow{\bar{x}} Q}{P' \parallel Q' \to P \parallel Q}$$

Definition 18 (Bisimulation)**.** A *bisimulation* on the set of processes is a binary relation b such that for all processes P and Q and all $x \in Act$, we have that PbQ implies

- if $P \xrightarrow{x} P'$, then there is a process Q' such that $Q \xrightarrow{x} Q'$ and $P'bQ'$,
- if $Q \xrightarrow{x} Q'$, then there is a process P' such that $P \xrightarrow{x} P'$ and $P'bQ'$.

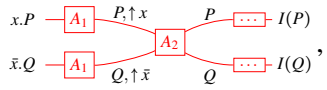
The *largest bisimulation* is the union of all bisimulations.

Labelled transitions define the *LTS semantics*, which, similarly to the reduction semantics, can be modelled as a monoidal category. Thus let **LTS** be the free monoidal category whose generating objects are pairs $P, \uparrow x$, where P is a process and $x \in Act$. We think of $\uparrow x$ as the "pending action", and omit the silent pending action: $P \coloneqq P, \uparrow \tau$. The morphisms of **LTS** are generated by



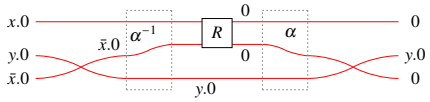
where P and Q range over processes, $x \in Act$ and $y \in A \cup \bar{A}$ and we identify $\bar{\bar{y}} \coloneqq y$. The structural isomorphisms of the monoidal category have the same form as the structural isomorphisms of **Red**, and correspond to the largest bisimulation. The other morphisms in **LTS** model those rewrite rules of the usual LTS semantics that are derivable via our restricted set of deduction rules.

There is a monoidal functor $I : \mathbf{Red} \to \mathbf{LTS}$, whose action on objects is defined as $0 \mapsto 0, \uparrow \tau$, $x.P \mapsto P, \uparrow \tau$ and $P \parallel Q \mapsto (I(P), I(Q))$. For morphisms, I takes each structural isomorphism in **Red** to the corresponding isomorphism in **LTS**, and the morphism R to

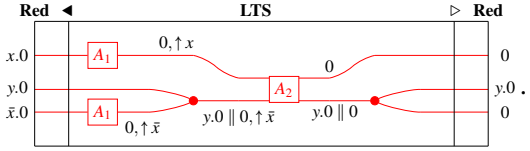


where the dots refer to an appropriate decomposition of P and Q into $I(P)$ and $I(Q)$.

We use the functor I to view the LTS semantics as the lower level language that explains the reduction semantics. For instance, we can explain the rewrite rule (16) by just moving its derivation in **Red**



through the window, that is, essentially by applying I. In this case, we are also able to give a counterfactual explanation:



The above diagram is indeed a counterfactual explanation (see the discussion in Section 4) of the rewrite rule (16): (1) the rewrite rule is an internal morphism in **Red**, (2) every non-identity internal morphism in the diagram is contained in **LTS**, which is strictly below **Red** in the partial order of the layered prop, (3) there are no 2-cells between the rewrite rule and the diagram. To see that (3) is indeed the case, note that there are in fact no 2-cells having the above diagram as either domain or codomain (one can see this by going through the generators of 2-cells of a layered prop one by one).

The fact that there is a counterfactual explanation of the rewrite rule (16) shows that it is not *necessary* to invoke the (analogue of) rule R in its derivation at the level of LTS. This observation allows us to show neatly that LTS semantics is more flexible than the reduction semantics, in the sense that there are more derivations of the same transitions. Note that the counterfactual explanation does not need to be more complex than an ordinary explanation: in this case it is in fact more direct, in the sense that it shows that there is an actual labelled transition, while the explanation obtained by translating the diagram in **Red** merely shows that there is a labelled transition up to the largest bisimulation.

8 Conclusions and Future Work

We have taken the first steps towards developing a mathematical framework for formalising explanations. Explanations in a category theoretic context usually take the form of a functor, whose domain is thought of as syntax and codomain as semantics. Our approach differs from this: in a layered prop, there are several possible translations to different levels, which are nonetheless syntactically represented in the same language (that is, within one layered prop). A layered prop allows one to easily work with different theories describing the same phenomenon, and, importantly, allows for partial translations instead of having to translate the full diagram, as we have illustrated with the examples. We have also observed

how counterfactual processes arise naturally within layered props: these are those processes that "look like" a translation without being one. Furthermore, the examples show that the same abstract principles hold in areas as distant as biology, electrical circuit theory and concurrency theory. Layered props can thus indeed be conceived as the initial stage of a general mathematical theory of explanations.

On the mathematical level, the next phase of developing the theory is to explore the precise connection of layered props to pointed profunctors. Currently, there is a canonical 2-functor which translates a layered prop generated by a system of monoidal categories to the category of pointed profunctors which preserves the axioms of a layered prop. One way to proceed would be to characterise the image of this functor, thus identifying a subcategory of $\mathbf{Prof}_*$ to which a given layered prop is equivalent. Another mathematical aspect that is important for practical applications is to modify the definition of a layered prop to allow for non-strictly associative monoidal categories, as for instance described diagrammatically in [23]. As briefly remarked in Section 6, the current semantics cannot adequately handle the important special case when the translation functor is faithful. This suggests that the current interpretation of the 2-cells as natural transformations is too restrictive, and some other notion of 2-cells for pointed profunctors should be used. In order to connect layered props to known structures, it would also be useful to express them as a Grothendieck construction.

Even though it was beyond the scope of this paper, we believe it is important to connect our work with the philosophy of science literature on explanations. Since the initial motivation for our work comes from biology, it is particularly interesting to see how ideas on explanations and causality in biology fit our framework. For instance, one of the main motivations of Robert Rosen for introducing the theoretical framework of *relational biology* was to put the *function* of an organism on equal grounding with the *mechanism* that underlies it [21]. This can be modelled within a layered prop: reductive and functional explanations are *a priori* completely symmetric, and in any case equally well-defined.

Several systems with multiple layers are known in the applied category theory literature. In addition to the already discussed [23] and [3] (Section 6), we mention the formalism of *hierarchical petri nets* [7], and Román's notion of an *open diagram* [20]. All of these rely on an intuitive notion composing processes at different levels, and hence we plan to explore them using layered props.

Acknowledgements The credit for the original idea for the "calculus of refinement and coarsening", as well as for the examples featuring chemical reactions and CCS goes to Jean Krivine. We also thank him for feedback and inspiring discussions during various stages of this work. We thank Samson Abramsky for discussing and giving feedback on ideas that lead to this work. We thank Cole Comfort for conversations and for pointing us towards the literature on internal string diagrams. LL thanks Jamie Vicary, Nick Hu, Alex Rice, Calin Tataru and Ioannis Markakis for an opportunity to present and discuss an early version of this work.

References

[1] John C. Baez & Brendan Fong (2015): *A Compositional Framework for Passive Linear Networks*, doi:10.48550/ARXIV.1504.05625. Available at `https://arxiv.org/abs/1504.05625`.

[2] Bruce Bartlett, Christopher L. Douglas, Christopher J. Schommer-Pries & Jamie Vicary (2015): *Modular categories as representations of the 3-dimensional bordism 2-category*, doi:10.48550/ARXIV.1509.06811. Available at `https://arxiv.org/abs/1509.06811`.

[3] Guillaume Boisseau & Paweł Sobociński (2021): *String Diagrammatic Electrical Circuit Theory*, doi:10.48550/ARXIV.2106.07763. Available at `https://arxiv.org/abs/2106.07763`.

[4] Filippo Bonchi, Robin Piedeleu, Pawel Sobocinski & Fabio Zanasi (2019): *Graphical Affine Algebra*. In: *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, IEEE, pp. 1–12, doi:10.1109/LICS.2019.8785877. Available at https://doi.org/10.1109/LICS.2019.8785877.

[5] Filippo Bonchi, Paweł Sobociński & Fabio Zanasi (2021): *A Survey of Compositional Signal Flow Theory*, pp. 29–56. Springer International Publishing, Cham, doi:10.1007/978-3-030-81701-5˙2. Available at https://doi.org/10.1007/978-3-030-81701-5_2.

[6] Francis Borceux (1994): *Bicategories and distributors*, p. 281–324. *Encyclopedia of Mathematics and its Applications* 1, Cambridge University Press, doi:10.1017/CBO9780511525858.009.

[7] Fabrizio Romano Genovese, Jelle Herold, Fosco Loregian & Daniele Palombi (2021): *A Categorical Semantics for Hierarchical Petri Nets*. *Electronic Proceedings in Theoretical Computer Science* 350, pp. 51–68, doi:10.4204/eptcs.350.4. Available at https://doi.org/10.4204%2Feptcs.350.4.

[8] William S. Hlavacek, James R. Faeder, Michael L. Blinov, Richard G. Posner, Michael Hucka & Walter Fontana (2006): *Rules for modeling signal-transduction systems*. *Science's STKE : signal transduction knowledge environment* 2006(344), p. re6, doi:10.1126/stke.3442006re6.

[9] Nick Hu (2019): *External traced monoidal categories*. Master's thesis, Department of Computer Science. Available at https://www.cs.ox.ac.uk/publications/publication13596-abstract.html.

[10] Jean Krivine (2017): *Systems Biology*. *ACM SIGLOG News* 4(3), p. 43–61, doi:10.1145/3129173.3129182. Available at https://doi.org/10.1145/3129173.3129182.

[11] Jean Krivine (2019): *Physical systems, composite explanations and diagrams*. https://youtu.be/w8vTTvA0HTE and https://www.cl.cam.ac.uk/events/syco/strings3-syco5/.

[12] Stephen Lack (2004): *Composing PROPs*. *Theory and Applications of Categories* 13(9), pp. 147–163.

[13] Jonathan Laurent, Jean Yang & Walter Fontana (2018): *Counterfactual Resimulation for Causal Analysis of Rule-Based Models*. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, International Joint Conferences on Artificial Intelligence Organization, pp. 1882–1890, doi:10.24963/ijcai.2018/260. Available at https://doi.org/10.24963/ijcai.2018/260.

[14] Fosco Loregian (2015): *Coend calculus*, doi:10.48550/ARXIV.1501.02503. Available at https://arxiv.org/abs/1501.02503.

[15] Saunders Maclane (1965): *Categorical Algebra*. *Bulletin (new series) of the American Mathematical Society* 71(1), pp. 40–106.

[16] Paul-André Melliès (2006): *Functorial Boxes in String Diagrams*. In Zoltán Ésik, editor: *Computer Science Logic*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–30, doi:10.1007/11874683˙1.

[17] Robin Milner (1980): *A Calculus of Communicating Systems*. *Lecture Notes in Computer Science* 92, Springer Netherlands, Netherlands, doi:10.1007/3-540-10235-3.

[18] Judea Pearl (2009): *The Logic of Structure-Based Counterfactuals*, 2 edition, p. 201–258. Cambridge University Press, doi:10.1017/CBO9780511803161.009.

[19] (2019): *Explainable AI: the Basics - Policy Briefing*. Available at royalsociety.org/ai-interpretability.

[20] Mario Romá n (2021): *Open Diagrams via Coend Calculus*. *Electronic Proceedings in Theoretical Computer Science* 333, pp. 65–78, doi:10.4204/eptcs.333.5. Available at https://doi.org/10.4204%2Feptcs.333.5.

[21] Robert Rosen (1991): *Life itself : a comprehensive inquiry into the nature, origin, and fabrication of life / Robert Rosen*. Complexity in ecological systems series, Columbia University Press, New York.

[22] P. Selinger (2010): *A Survey of Graphical Languages for Monoidal Categories*. In: *New Structures for Physics*, Springer Berlin Heidelberg, pp. 289–355, doi:10.1007/978-3-642-12821-9˙4. Available at https://doi.org/10.1007%2F978-3-642-12821-9_4.

[23] Paul Wilson, Dan Ghica & Fabio Zanasi (2022): *String diagrams for non-strict monoidal categories*, doi:10.48550/ARXIV.2201.11738. Available at `https://arxiv.org/abs/2201.11738`.

[24] Fabio Zanasi (2015): *Interacting Hopf Algebras: the theory of linear systems*. Ph.D. thesis, Laboratoire de l'Informatique du Parallélisme, doi:10.48550/ARXIV.1805.03032. Available at `https://arxiv.org/abs/1805.03032`.

A Profunctors and Pointed Profunctors

In order to fix notational conventions, we recall the standard definition of profunctors. We also define the not-so-standard category of pointed profunctors. We state the results about (pointed) profunctors needed in the main body of the paper, mostly without proof.

A.1 Profunctors

We follow Loregian [14] in our discussion of profunctors and coends.

Definition 19 (Bicategory of profunctors)**.** Define the bicategory of *profunctors* **Prof** as follows.

- the 0-cells are (small) categories,
- the 1-cells, denoted by $\mathcal{C} \nrightarrow \mathcal{D}$, are functors

$$\mathcal{C}^{op} \times \mathcal{D} \to \mathbf{Set},$$

- the $2-cells$ are natural transformations $\alpha : F \Rightarrow G$,
- the composition

$$c_{\mathcal{A},\mathcal{B},\mathcal{C}} : \mathbf{Prof}(\mathcal{A},\mathcal{B}) \times \mathbf{Prof}(\mathcal{B},\mathcal{C}) \to \mathbf{Prof}(\mathcal{A},\mathcal{C})$$

takes profunctors $F : \mathcal{A} \nrightarrow \mathcal{B}$ and $G : \mathcal{B} \nrightarrow \mathcal{C}$ to the coend $G \circ F = \int^B F(-,B) \times G(B,=)$. Explicitly, we define

$$(G \circ F)(A,C) \coloneqq \int^{B \in \mathcal{B}} F(A,B) \times G(B,C).$$

There is a bifunctor

$$\times : \mathbf{Prof} \times \mathbf{Prof} \to \mathbf{Prof}$$

defined as the product functor of n-cells for each $n = 0,1,2$ which equips **Prof** with a symmetric monoidal structure.

Given a 2-category $\mathcal{K}$, let us write $\mathcal{K}^{op}$ for the 2-category whose 0-cells and 2-cells are those of $\mathcal{K}$ and whose 1-cells are the reversed 1-cells of $\mathcal{K}$, that is, for all 0-cells A and B we have $\mathcal{K}^{op}(A,B) = \mathcal{K}(B,A)^{op}$. Similarly, we write $\mathcal{K}^{co}$ for the 2-category whose 0-cells and 1-cells are those of $\mathcal{K}$ and whose 2-cells are the reversed 2-cells of $\mathcal{K}$, that is, for all 0-cells A and B we have $\mathcal{K}^{co}(A,B) = \mathcal{K}(A,B)^{op}$.

There are two ways to embed the 2-category **Cat** into **Prof**: one is contravariant on the 1-cells, the other on the 2-cells. Both embeddings are identity on objects. The embedding

$$\mathfrak{p}^- : \mathbf{Cat}^{co} \to \mathbf{Prof}$$

takes a functor $F : \mathcal{C} \to \mathcal{D}$ to the profunctor $\mathfrak{p}^F : \mathcal{C} \nrightarrow \mathcal{D}$ defined on objects by $\mathfrak{p}^F(C,D) \coloneqq \mathcal{D}(FC,D)$, and a natural transformation $\eta : F \to G$ to the natural transformation $\mathfrak{p}^G \to \mathfrak{p}^F$ whose (C,D)-component is given by $- \circ \eta_C$.

Dually, the embedding

$$\mathfrak{p}_- : \mathbf{Cat}^{op} \to \mathbf{Prof}$$

takes a functor $F : \mathcal{C} \to \mathcal{D}$ to the profunctor $\mathfrak{p}_F : \mathcal{D} \nrightarrow \mathcal{C}$ defined on objects by $\mathfrak{p}_F(C,D) \coloneqq \mathcal{D}(D,FC)$, and a natural transformation $\eta : F \to G$ to the natural transformation $\mathfrak{p}^F \to \mathfrak{p}^G$ whose (C,D)-component is given by $\eta_C \circ -$.

Both $\mathfrak{p}^-$ and $\mathfrak{p}_-$ are 2-functors, locally fully faithful and for every functor F the 1-cell $\mathfrak{p}^F$ is the left adjoint to $\mathfrak{p}_F$ in the bicategory **Prof** (see section 5.1 of Loregian [14] for the details).

Proposition 20. *Both* $\mathfrak{p}^- : \mathbf{Cat}^{co} \to \mathbf{Prof}$ *and* $\mathfrak{p}_- : \mathbf{Cat}^{op} \to \mathbf{Prof}$ *are monoidal 2-functors.*

Proof. We prove the result for $\mathfrak{p}^-$: the argument for $\mathfrak{p}_-$ is dual.

Since the embedding is identity on objects, the monoidal product of 0-cells (which is just the cartesian product of categories) is preserved.

For 1-cells, let $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{C}' \to \mathcal{D}'$ be functors. We wish to show that $\mathfrak{p}^{F\times G} \simeq \mathfrak{p}^F \times \mathfrak{p}^G$. We compute as follows:

$$\begin{aligned}
\mathfrak{p}^{F\times G}(C,C';D,D') &= \mathcal{D}\times\mathcal{D}'(F\times G(C,C'),(D,D')) \\
&= \mathcal{D}\times\mathcal{D}'((FC,GC'),(D,D')) \\
&= \mathcal{D}(FC,D)\times\mathcal{D}'(GC',D') \\
&= \mathfrak{p}^F(C,D)\times\mathfrak{p}^G(C',D') \\
&= \left(\mathfrak{p}^F\times\mathfrak{p}^G\right)(C,C';D,D'),
\end{aligned}$$

whence it follows that $\mathfrak{p}^{F\times G}$ and $\mathfrak{p}^F \times \mathfrak{p}^G$ agree on objects. The fact that they agree on morphisms is a similar computation.

For 2-cells, let $F,F' : \mathcal{C} \to \mathcal{D}$ and $G,G' : \mathcal{C}' \to \mathcal{D}'$ all be functors. Given natural transformations $\eta : F \to F'$ and $\mu : G \to G'$, we wish to show that $\mathfrak{p}^{\eta\times\mu} = \mathfrak{p}^\eta \times \mathfrak{p}^\mu$. This follows by observing that their components coincide:

$$\mathfrak{p}^{\eta\times\mu}_{(C,C';D,D')} = -\circ(\eta\times\mu)_{(C,C')} = (-\circ\eta_C)\times(-\circ\mu_{C'}) = \mathfrak{p}^\eta_{C,D}\times\mathfrak{p}^\mu_{C',D'} = (\mathfrak{p}^\eta\times\mathfrak{p}^\mu)_{C,C';D,D'}.$$

□

A.2 Pointed Profunctors

Definition 21 (Pointed profunctors)**.** Define the bicategory of *pointed profunctors* $\mathbf{Prof}_*$ as follows:

- the 0-cells are pairs $(\mathcal{C},c)$ of a (small) category $\mathcal{C}$ and an object $c \in \mathcal{O}b(\mathcal{C})$,
- the 1-cells $(P,f) : (\mathcal{C},c) \to (\mathcal{D},d)$ consist of a profunctor $P : \mathcal{C} \nrightarrow \mathcal{D}$, that is, a functor

  $$P : \mathcal{C}^{op}\times\mathcal{D} \to \mathbf{Set},$$

  together with an element $f \in P(c,d)$,
- the 2-cells $\alpha : (P,f) \to (Q,g)$ are natural transformations $\alpha : P \Rightarrow Q$ such that $\alpha_{c,d}(f) = g$,
- the composition of $(P,f) : (\mathcal{C},c) \to (\mathcal{D},d)$ and $(Q,g) : (\mathcal{D},d) \to (\mathcal{E},e)$ is given by $(Q\circ P,[f,g])$, where $\circ$ is the composition of profunctors and $[f,g]$ the equivalence class of the pair (f,g) in $(Q\circ P)(c,e)$.

Note that a pointed hom-functor $(\mathcal{C}(-,-),f) : (\mathcal{C},c) \to (\mathcal{C},c')$ is precisely a morphism $f : c \to c'$. Thus we will simply write the hom-functor $(\mathcal{C}(-,-),f)$ as f. For a category $\mathcal{C}$, we define an assignment

$$\begin{aligned}
z_{\mathcal{C}} : \mathcal{C} &\to \mathbf{Prof}_* \\
c &\mapsto (\mathcal{C},c) \\
(f : c \to c') &\mapsto (f : (\mathcal{C},c) \to (\mathcal{C},c'))
\end{aligned}$$

Proposition 22. *The assignment $z_{\mathcal{C}}$ is a pseudofunctor (when $\mathcal{C}$ is taken to have the trivial bicategory structure).*

Proof. We first show that $z_{\mathcal{C}}$ preserves composition. Thus let $f : c \to d$ and $g : d \to e$ be morphisms in $\mathcal{C}$. First, $\mathcal{C}(-,-) \circ \mathcal{C}(-,-) \simeq \mathcal{C}(-,-)$ since the hom-profunctor is the identity profunctor. Observe that such an isomorphism is given by $\int^a \mathcal{C}(c,a) \times \mathcal{C}(a,e) \xrightarrow{\sim} \mathcal{C}(c,e)$ given by $[n,m] \mapsto m \circ n$ (this is well-defined). Thus in particular $[f,g] \mapsto gf$, whence

$$z_{\mathcal{C}}(g) \circ z_{\mathcal{C}}(f) = (\mathcal{C}(-,-),g) \circ (\mathcal{C}(-,-),f) \simeq (\mathcal{C}(-,-),gf) = z_{\mathcal{C}}(gf).$$

From the above it follows that $\mathrm{id}_c : (\mathcal{C},c) \to (\mathcal{C},c)$ is the identity on $(\mathcal{C},c)$, so that $z_{\mathcal{C}}$ preserves the identities. □

There is a pseudofunctor

$$\times : \mathbf{Prof}_* \times \mathbf{Prof}_* \to \mathbf{Prof}_*$$

defined as

- $(C,c) \times (D,d) \coloneqq (C \times D, (c,d))$ on the 0-cells,
- $(P,f) \times (Q,g) \coloneqq (P \times Q, (f,g))$ on the 1-cells,
- the product of natural transformations on the 2-cells.

Writing 1 for the terminal category and $\bullet$ for its unique object, we have the following:

Proposition 23. $(\mathbf{Prof}_*, \times, (1,\bullet))$ *is a symmetric monoidal bicategory.*

B Semantic Properties of Layered Props

We discuss the properties that the interpretation functor $\mathcal{I} : \mathcal{L}(\Omega) \to \mathbf{Prof}_*$ has. Throughout the section, we assume that Ω is a system of monoidal categories.

We begin by proving that $\mathcal{I}$ is indeed a pointed profunctor model (Proposition 5).

Proof of Proposition 5. The equalities of morphisms for each category $\omega \in \Omega$ are preserved by Proposition 22. The unit and counit maps in Figure 3 are preserved and the triangle equalities for them hold since we have defined each pair of profunctors as an adjoint pair. Since all the internal morphisms are identities, there is nothing to show for the composition of the internal morphisms.

All the rules in Figure 4 follow from the fact that each category and functor in Ω is monoidal and that both $\mathfrak{p}^-$ and $\mathfrak{p}_-$ are monoidal 2-functors (Proposition 20). For example, by (strict) associativity we have that $\otimes(\mathrm{id} \times \otimes) = \otimes(\otimes \times \mathrm{id})$ in **Cat**. We get the desired equations by applying the embeddings:

$$\mathfrak{p}^{\otimes} \circ (\mathfrak{p}^{\otimes} \times \mathrm{id}) = \mathfrak{p}^{\otimes} \circ (\mathrm{id} \times \mathfrak{p}^{\otimes}) \quad \texttt{assoc},$$
$$(\mathrm{id} \times \mathfrak{p}_{\otimes}) \circ \mathfrak{p}_{\otimes} = (\mathfrak{p}_{\otimes} \times \mathrm{id}) \circ \mathfrak{p}_{\otimes} \quad \texttt{coassoc}.$$

It remains to show that the rules in Figure 2 are preserved. These are the only rules with a non-trivial internal structure. Observe that all these rules are either of the form

$$(\mathfrak{p}^f, \mathrm{id}_{f\beta}) \circ \sigma \simeq f\sigma \circ (\mathfrak{p}^f, \mathrm{id}_{f\alpha}) \qquad \text{or} \qquad \sigma \circ (\mathfrak{p}_f, \mathrm{id}_{f\beta}) \simeq (\mathfrak{p}_f, \mathrm{id}_{f\alpha}) \circ f\sigma$$

for some functor $f : \omega \to \tau$ and some morphism $\sigma : \alpha \to \beta$. We show the isomorphism on the left. First, at the level of profunctors the isomorphism holds since hom-functors are the identities. It remains to show

that $[\sigma, \mathrm{id}_{f\beta}] \sim [\mathrm{id}_{f\alpha}, f\sigma]$ under this isomorphism. To this end, note that the left-hand side evaluates via the isomorphism

$$\int^{\gamma \in \omega} \omega(\alpha,\gamma) \times \omega(f\gamma, f\beta) \simeq \omega(f\alpha, f\beta) \qquad [h,k] \mapsto k \circ fh$$

to $f\sigma$. Similarly, the right-hand side evaluates via the isomorphism

$$\int^{\gamma \in \tau} \omega(f\alpha,\gamma) \times \omega(\gamma, f\beta) \simeq \omega(f\alpha, f\beta) \qquad [h,k] \mapsto k \circ h$$

also to $f\sigma$, whence the desired identification follows. The argument for the rules of the second form is dual. □

The following proposition shows that we can detect properties of monoidal categories in a layered prop. This observation is not relevant for the examples that we discuss in this work, yet it is important for the development of the general theory of layered props.

Proposition 24. *If both $\tau, \omega \in \Omega$ are monoidal closed (*resp. coclosed*) and $f : \omega \to \tau$ in Ω is also monoidal closed (*resp. coclosed*), then the interpretation $\mathcal{I}$ preserves the* 2*-cell* `C` (resp. `coC`) *in Figure 8.*

Proof. For `C`, we have to show that

$$\mathfrak{p}^{\otimes} \circ (\mathfrak{p}_f \times \mathrm{id}) \simeq \mathfrak{p}_f \circ \mathfrak{p}^{\otimes} \circ (\mathrm{id} \times \mathfrak{p}^f).$$

Both profunctors are of the type $\tau \times \omega \nrightarrow \omega$. Let us compute both sides on the triple of objects (D, C, C'). The right-hand side computes to

$$\int^{B,E \in \tau} \tau(fC, B) \times \tau(D \otimes B, E) \times \tau(E, FC') \simeq \tau(D \otimes FC, FC'),$$

while in order to reduce the left-hand side we use the monoidal closed structure:

$$\begin{aligned} \int^{A \in \omega} \tau(D, FA) \times \omega(A \otimes C, C') &\simeq \int^{A \in \omega} \tau(D, FA) \times \omega(A, [C, C']) \\ &\simeq \tau(D, F[C, C']) \\ &\simeq \tau(D, [FC, FC']) \\ &\simeq \tau(D \otimes FC, FC'). \end{aligned}$$

Since these agree and all isomorphisms are natural, we have the desired isomorphism. The argument for `coC` is dual, using that the categories and functors are monoidal coclosed. □

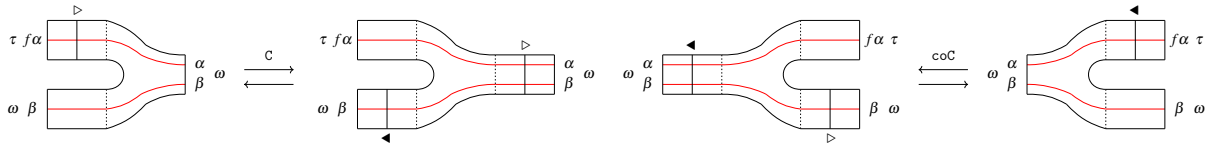


Figure 8: Monoidal (co)closure equations.