



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

ARCHIVIO ISTITUZIONALE
DELLA RICERCA

Alma Mater Studiorum Università di Bologna Archivio istituzionale della ricerca

Bialgebraic foundations for the operational semantics of string diagrams

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

Published Version:

Bialgebraic foundations for the operational semantics of string diagrams / Bonchi F.; Piedeleu R.; Sobocinski P.; Zanasi F.. - In: INFORMATION AND COMPUTATION. - ISSN 0890-5401. - ELETTRONICO. - 281:(2021), pp. 104767.1-104767.33. [10.1016/j.ic.2021.104767]

Availability:

This version is available at: <https://hdl.handle.net/11585/904705> since: 2022-11-21

Published:

DOI: <http://doi.org/10.1016/j.ic.2021.104767>

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>).
When citing, please refer to the published version.

(Article begins on next page)

This is the final peer-reviewed accepted manuscript of:

Filippo Bonchi, Robin Piedeleu, Paweł Sobociński, Fabio Zanasi, Bialgebraic foundations for the operational semantics of string diagrams, Information and Computation, Volume 281, 2021, 104767, ISSN 0890-5401

The final published version is available online at:
<https://dx.doi.org/10.1016/j.ic.2021.104767>

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>)

When citing, please refer to the published version.

Bialgebraic Foundations for the Operational Semantics of String Diagrams

Filippo Bonchi¹, Robin Piedeleu¹, Paweł Sobociński¹, Fabio Zanasi¹

^a *Università di Pisa, Italy*

^b *University College London, United Kingdom*

^c *Tallinn University of Technology, Estonia*

Abstract

Turi and Plotkin’s bialgebraic semantics is an abstract approach to specifying the operational semantics of a system, by means of a distributive law between its syntax (encoded as a monad) and its dynamics (an endofunctor). This setup is instrumental in showing that a semantic specification (a coalgebra) satisfies desirable properties: in particular, that it is compositional.

In this work, we use the bialgebraic approach to derive well-behaved structural operational semantics of *string diagrams*, a graphical syntax that is increasingly used in the study of interacting systems across different disciplines. Our analysis relies on representing the two-dimensional operations underlying string diagrams in various categories as a monad, and their bialgebraic semantics in terms of a distributive law for that monad.

As a proof of concept, we provide bialgebraic compositional semantics for a versatile string diagrammatic language which has been used to model both signal flow graphs (control theory) and Petri nets (concurrency theory). Moreover, our approach reveals a correspondence between two different interpretations of the Frobenius equations on string diagrams and two synchronisation mechanisms for processes, à la Hoare and à la Milner.

Keywords: String Diagram, Structural Operational Semantics, Bialgebraic semantics

1. Introduction

Starting from the seminal works of Hoare and Milner, there was an explosion [? ? ? ?] of interest in process calculi: formal languages for specifying and reasoning about concurrent systems. The beauty of the approach, and often the focus of research, lies in *compositionality*: essentially, the behaviour of composite systems—usually understood as some kind of process equivalence, the

*Supported by the ESF funded Estonian IT Academy research measure (project 2014-2020.4.05.19-0001)

7 most famous of which is bisimilarity—ought to be a function of the behaviour
8 of its components. The central place of compositionality led to the study of syn-
9 tactic formats for semantic specifications [? ? ?]; succinctly stated, syntactic
10 operations with semantics defined using such formats are homomorphic wrt the
11 semantic space of behaviours.

12 Another thread of concurrency theory research [? ? ?] uses graphical
13 formalisms, such as Petri nets. These often have the advantage of highlighting
14 connectivity, distribution and the communication topology of systems. They
15 tend to be popular with practitioners in part because of their intuitive and
16 human-readable depictions, an aspect that should not be underestimated: in-
17 deed, pedagogical texts introducing CCS [?] and CSP [?] often resort to
18 pictures that give intuition about topological aspects of syntactic specifications.
19 However, compositionality has not—historically—been a principal focus of re-
20 search.

21 In this paper we propose a framework that allows us to eat our cake and have
22 it too. We use *string diagrams* [?] which have an intuitive graphical rendering,
23 but also come with algebraic operations for composition. String diagrams com-
24 bine the best of both worlds: they are a (2-dimensional) syntax, but also convey
25 important topological information about the systems they specify. They have
26 been used in recent years to give compositional accounts of quantum circuits[?
27 ?], signal flow graphs [? ? ?], Petri nets [?], and electrical circuits [? ?],
28 amongst several other applications.

29 Our main contribution is the adaptation of Turi and Plotkin’s bialgebraic
30 semantics (*abstract GSOS*) [? ?] for string diagrams. By doing so, we provide
31 a principled justification and theoretical framework for giving definitions of op-
32 erational semantics to the generators and operations of string diagrams, which
33 are those of monoidal categories. More precisely we deal with string diagrams
34 for symmetric monoidal categories which organise themselves as arrows of a
35 particularly simple and well-behaved class known as *props*. Similar operational
36 definitions have been used in the work on the algebra of $\text{Span}(\text{Graph})$ [?], tile
37 logic [?], the wire calculus [?] and recent work on modelling signal flow graphs
38 and Petri nets [? ?]. In each case, semantics was given either monolithically or
39 via a set of SOS rules. The link with bialgebraic framework—developed in this
40 paper—provides us a powerful theoretical tool that (i) justifies these operational
41 definitions and (ii) guarantees compositionality.

In a nutshell, in the bialgebraic approach, the syntax of a language is the ini-
tial algebra (the algebra of terms) T_Σ for a signature functor Σ . A certain kind
of distributive law, an *abstract GSOS* specification [?], induces a coalgebra (a
state machine) $\beta: T_\Sigma \rightarrow \mathcal{F}T_\Sigma$ capturing the operational semantics of the lan-
guage. The final \mathcal{F} -coalgebra Ω provides the denotational universe: intuitively,
the space of all possible behaviours. The unique coalgebra map $\llbracket \cdot \rrbracket_\beta: T_\Sigma \rightarrow \Omega$

represents the denotational semantics assigning to each term its behaviour.

$$\begin{array}{ccc}
 T_\Sigma & \xrightarrow{\llbracket \cdot \rrbracket_\beta} & \Omega \\
 \beta \downarrow & & \downarrow \\
 \mathcal{F}(T_\Sigma) & \xrightarrow{\mathcal{F}(\llbracket \cdot \rrbracket_\beta)} & \mathcal{F}(\Omega)
 \end{array} \tag{1}$$

42 The crucial observation is that (??) lives in the category of Σ -algebras: Ω also
 43 carries a Σ -algebra structure and the denotational semantics is an algebra homo-
 44 morphism. This means that the behaviour of a composite system is determined
 45 by the behaviour of the components, e.g. $\llbracket s + t \rrbracket = \llbracket s \rrbracket + \llbracket t \rrbracket$, for an operation $+$
 46 in Σ .

47 We show that the above framework can be adapted to the algebra of string
 48 diagrams. The end result is a picture analogous to (??), but living in the
 49 category of props and prop morphism. As a result, the denotational map is a
 50 prop morphism, and thus guarantees compositionality with respect to sequential
 51 and parallel composition of string diagrams.

52 Adapting the bialgebraic approach to the 2-dimensional syntax of props
 53 requires some technical work since, e.g. the composition operation of monoidal
 54 categories is a dependent operation. For this reason we need to adapt the usual
 55 notion of a syntax endofunctor on the category of sets; instead we work in a
 56 category \mathbf{Sig} whose objects are spans $\mathbb{N} \leftarrow \Sigma \rightarrow \mathbb{N}$, with the two legs giving the
 57 number of dangling wires on the left and right of each diagram. The operations
 58 of props are captured as a \mathbf{Sig} -endofunctor, which yields string-diagrams-as-
 59 initial-algebra, and a quotient of the resulting free monad, whose algebras are
 60 precisely props.

61 In addition to the basic laws of props, we also consider the further imposition
 62 of the equations of special Frobenius algebras. We illustrate the role of this
 63 algebraic structure with our running example, a string diagrammatic process
 64 calculus \mathbf{Circ}_R that has two Frobenius structures and can be equipped with two
 65 different semantics, one which provides a compositional account of signal flow
 66 graphs for linear time invariant dynamical systems [?], and one which is a
 67 compositional account of Petri nets [?].

68 We conclude with an observation that ties our work back to classical concepts
 69 of process calculi and show that the two Frobenius structures of \mathbf{Circ}_R are closely
 70 related to two different, well-known synchronisation patterns, namely those of
 71 Hoare's CSP [?] and Milner's CCS [?].

72 *Structure of the paper.* In §?? we introduce our main example and recall
 73 some preliminaries, followed by a recapitulation of bialgebraic approach in §??.
 74 We develop the technical aspects of string-diagrams-as-syntax in §?? and adapt
 75 the bialgebraic approach in §??. Finally, we exhibit the connection with classical
 76 synchronisation mechanisms in §?? and conclude in §??.

77 This work extends the conference paper [?] with a greatly expanded section
 78 on preliminaries (Section ??), ??, and all the missing proofs. Moreover,
 79 Section ??, Remark ?? and Section ?? are new contributions.

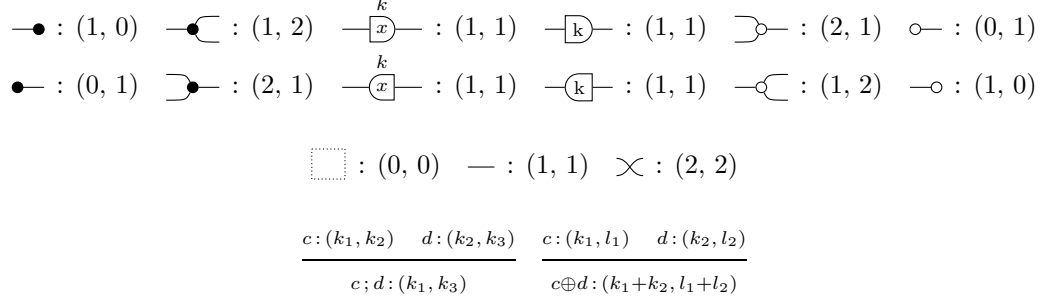


Figure 1: Sorting discipline for $\text{Circ}_{\mathbb{R}}$

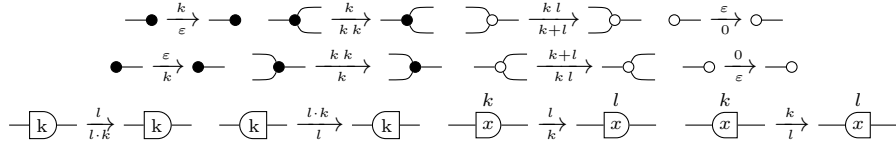


Figure 2: Structural Operational Semantics for the generators of $\text{Circ}_{\mathbb{R}}$. Intuitively, from left to right, these are elementary connectors modelling discard, copy, one-place register, multiplication by a scalar, addition, and the constant zero.

80 2. Motivating Example

As our motivating example, we recall from [? ? ?] a basic language $\text{Circ}_{\mathbb{R}}$ given by the grammar below. Values k in $-\overset{k}{x}$ and $-\boxed{k}$ range over elements of a given semiring \mathbb{R} .

$$c, d ::= -\bullet \mid -\bullet \frown \mid -\overset{k}{x} \mid -\boxed{k} \mid \frown\text{-} \mid \circ\text{-} \mid \bullet\text{-} \mid \frown\bullet \mid \text{-}\circ\text{-} \quad (2)$$

$$-\overset{k}{x} \mid -\boxed{k} \mid -\circ\text{-} \mid -\circ \quad (3)$$

$$\boxed{} \mid - \mid \times \mid c ; d \mid c \oplus d \quad (4)$$

81 The language does not feature variables; on the other hand, a simple sorting
82 discipline is necessary. A sort is a pair (n, m) , with $n, m \in \mathbb{N}$. Henceforth we
83 will consider only terms sortable according to the rules in Figure ???. An easy
84 induction confirms uniqueness of sorting.

85 The operational meaning of terms is defined recursively by the structural
86 rules in Figs. ?? and ?? where k, l range over \mathbb{R} and $\mathbf{a}, \mathbf{b}, \mathbf{c}$ over \mathbb{R}^* , the set of
87 words over \mathbb{R} . We denote the empty word by ε and concatenation of \mathbf{a}, \mathbf{b} by
88 \mathbf{ab} . As expected $+$, \cdot and 0 denote respectively the sum, the product and zero
89 of the semiring \mathbb{R} . For any term $c : (n, m)$, the rules yield a labelled transition
90 system where each transition has form $c \xrightarrow{\mathbf{a}} d$. By induction, it is immediate
91 that d has the same sort as c , the word \mathbf{a} has length n , and \mathbf{b} has length m .

$$\begin{array}{ccc}
\frac{c \xrightarrow{a} c' \quad d \xrightarrow{b} d'}{c; d \xrightarrow{a} c'; d'} \lambda^{\text{sq}} & & \frac{s \xrightarrow{a_1} c' \quad d \xrightarrow{a_2} d'}{c \oplus d \xrightarrow{a_1 a_2} d' \oplus d'} \lambda^{\text{mp}} \\
\frac{\quad}{\square \xrightarrow{\varepsilon} \square} \lambda^{\varepsilon} & \xrightarrow{\quad} \lambda^{\text{id}} & \frac{\quad}{\bowtie \xrightarrow{kl} \bowtie} \lambda^{\text{sy}} \\
\quad & \frac{\quad}{\text{---} \xrightarrow{k} \text{---}} & \quad
\end{array}$$

Figure 3: Structural operational semantics for the operations of $\text{Circ}_{\mathbb{R}}$.

92 Our chief focus in this paper is the study of semantics specifications of the
93 kind given in Figs. ?? and ?. So far, the technical difference with typical
94 GSOS examples [?] is the presence of a sorting discipline. A more significant
95 difference, which we will now highlight, is that sorted terms are considered up-to-
96 the laws of symmetric monoidal categories. As such, they are “2-dimensional
97 syntax” and enjoy a pictorial representation in terms of string diagrams.

98 2.1. From Terms to String Diagrams

In (??)-(??) we purposefully used a graphical rendering of the components. Indeed, terms of $\text{Circ}_{\mathbb{R}}$ are usually represented graphically, according to the con-

vention that $c; c'$ is drawn $\begin{array}{|c|} \hline c \\ \hline c' \\ \hline \end{array}$ and $c \oplus c'$ is drawn $\begin{array}{|c|} \hline c \\ \hline c' \\ \hline \end{array}$. For instance,

the term $((\bullet\text{---}; \text{---}\bullet) \oplus \text{---}); ((\text{---} \oplus \text{---}\bullet); \text{---}\bullet)$ is depicted as the following diagram.

$$p_k ::= \text{---} \bullet \text{---} \bullet \text{---} \begin{array}{|c|} \hline k \\ \hline x \\ \hline \end{array} \text{---} \bullet \text{---} \text{---} \quad (5)$$

99 Given this graphical convention, a sort gives the number of dangling wires on
100 each side of the diagram induced by a term. A transition $c \xrightarrow{a} d$ means that
101 c may evolve to d when the values on the dangling wires on the left are \mathbf{a} and
102 those on the right are \mathbf{b} . When \mathbb{R} is the natural numbers, the diagram in (??)
103 behaves as a place of a Petri nets containing k tokens: any number of tokens
104 can be inserted from its left and at most k tokens can be removed from its right.
105 Indeed, by the rules in Figs. ?? and ??, $p_k \xrightarrow{i} p_{k'}$ iff $o \leq k$ and $k' = i + k - o$.

The graphical notation is appealing as it highlights connectivity and the capability for resource exchange. However, syntactically different terms can yield the same diagram, e.g. $(\bullet\text{---}\oplus\text{---}); (\text{---}\bullet\text{---}\oplus\text{---}); (\text{---}\oplus\text{---}\bullet)$; $(\text{---}\oplus\text{---}\bullet)$; $(\text{---}\oplus\text{---}\bullet)$; $(\text{---}\oplus\text{---}\bullet)$ also yields (??). Indeed, one defines diagrams to be terms modulo *structural congruence*, denoted by \equiv . This is the smallest congruence over terms generated by the equations of strict symmetric monoidal

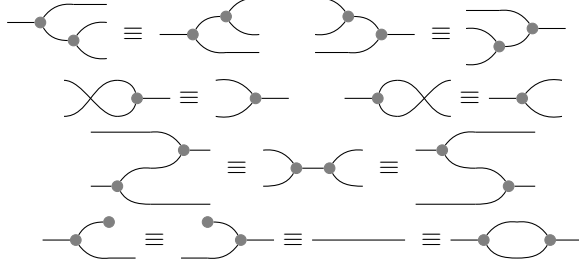


Figure 4: Axioms of special Frobenius bimonoids

categories (SMCs):

$$(f; g); h \equiv f; (g; h) \quad (f; id_m) \equiv f \quad (id_n; f) \equiv f \quad (6)$$

$$(f \oplus g) \oplus h \equiv f \oplus (g \oplus h) \quad (\epsilon \oplus f) \equiv f \quad (f \oplus \epsilon) \equiv f \quad (7)$$

$$(f; g) \oplus (h; i) \equiv (f \oplus h); (g \oplus i) \quad (8)$$

$$\sigma_{1,1}; \sigma_{1,1} \equiv id_2 \quad (\sigma_{1,n}; (f \oplus id_1)) \equiv (id_1 \oplus f); \sigma_{1,m} \quad (9)$$

$$(\sigma_{n,1}; (id_1 \oplus g)) \equiv (g \oplus id_1); \sigma_{m,1} \quad (10)$$

106 where identities $id_n : (n, n)$ and symmetries $\sigma_{n,m} : (n + m, m + n)$ can be
 107 recursively defined starting from $id_0 := \square$ and $\sigma_{1,1} := \bowtie$. Therefore, sorted
 108 diagrams $c : (n, m)$ are the arrows $n \rightarrow m$ of an SMC with objects the natural
 109 numbers, also called a prop [?].

110 2.2. Frobenius Bimonoids

111 We will also consider additional algebraic structure for the black (\bullet , \bullet , \bullet , \bullet),
 112 \bullet , \bullet) and the white (\circ , \circ , \circ , \circ) components. When \mathbb{R} is the field
 113 of reals, $\text{Circ}_{\mathbb{R}}$ models linear dynamical systems [? ? ?] and both the black and
 114 the white structures form special Frobenius bimonoids, meaning the axioms of
 115 Fig. ?? hold, replacing the gray circles by either black or white. When \mathbb{R} is the
 116 semiring of natural numbers, $\text{Circ}_{\mathbb{R}}$ models Petri nets [?] and only the black
 117 structure satisfies these equations. In § ??, we shall see that the black Frobenius
 118 structure gives rise to the synchronisation mechanism used by Hoare in CSP [?
 119], while the white Frobenius structure to that used by Milner in CCS [?].

120 3. Background: Bialgebras and GSOS Specifications

121 **(Co)algebras for endofunctors.** Given a category \mathcal{C} and a functor $\mathcal{F} : \mathcal{C} \rightarrow$
 122 \mathcal{C} , an \mathcal{F} -algebra is a pair (X, α) for an object X and an arrow $\alpha : \mathcal{F}X \rightarrow X$ in \mathcal{C} .
 123 An \mathcal{F} -algebra morphism $f : (X, \alpha) \rightarrow (X', \alpha')$ is an arrow $f : X \rightarrow X'$ such that
 124 $f \circ \alpha = \alpha' \circ \mathcal{F}f$. The category of \mathcal{F} -algebras and their morphisms is denoted by
 125 $\text{Alg}(\mathcal{F})$.

126 Coalgebras are defined dually: an \mathcal{F} -coalgebra is a pair (X, β) for an object X
 127 and an arrow $\alpha : X \rightarrow \mathcal{F}X$ in \mathcal{C} ; an \mathcal{F} -coalgebra morphism $f : (X, \beta) \rightarrow (X', \beta')$

128 is an arrow $f: X \rightarrow X'$ such that $\alpha' \circ f = \mathcal{F}f \circ \alpha$. The category of \mathcal{F} -coalgebras
 129 and their morphisms is denoted by $CoAlg(\mathcal{F})$.

130 When \mathcal{C} is **Set**, coalgebras can be thought as state-based systems: an \mathcal{F} -
 131 *coalgebra* (X, β) consists of a set of states X and a "transition" function $\beta: X \rightarrow$
 132 $\mathcal{F}X$. The functor \mathcal{F} define the type of the transitions: for instance, coalgebras
 133 for the functor $\mathcal{F}X = 2 \times X^A$ are deterministic automata (see ??). When a
 134 final object (Ω, ω) exists in $CoAlg(\mathcal{F})$, this can be thought as a universe of all
 135 possible \mathcal{F} -behaviours. The unique \mathcal{F} -coalgebra morphism $[\cdot]: (X, \beta) \rightarrow (\Omega, \omega)$
 136 is a function mapping every state of X into its behaviour.

137 **Monads.** An endofunctor \mathcal{T} forms a *monad* when it is equipped with natural
 138 transformations $\eta: Id \rightarrow \mathcal{T}$ and $\mu: \mathcal{T}\mathcal{T} \rightarrow \mathcal{T}$ such that $\mu \circ \eta\mathcal{T} = Id_{\mathcal{T}} = \mu \circ \mathcal{T}\eta$
 139 and $\mu \circ \mathcal{T}\mu = \mu \circ \mu\mathcal{T}$. An *Eilenberg-Moore algebra for a monad* (\mathcal{T}, η, μ) is a
 140 \mathcal{T} -algebra $\alpha: \mathcal{T}X \rightarrow X$ such that $\alpha \circ \eta_X = id_X$ and $g \circ \mu_X = g \circ \mathcal{T}g$. Morphisms
 141 are simply morphisms of \mathcal{T} -algebras. The category of Eilenberg-Moore algebras
 142 for \mathcal{T} and their morphisms is denoted by $EM(\mathcal{T})$. For the sake of brevity,
 143 when \mathcal{T} is a monad, \mathcal{T} -algebra will not mean algebra for the endofunctor \mathcal{T} ,
 144 but Eilenberg-Moore algebra for the monad \mathcal{T} .

145 A *monad morphism* from a monad (\mathcal{T}, η, μ) to a monad $(\mathcal{T}', \eta', \mu')$ on the
 146 same category \mathcal{C} is a natural transformation $\kappa: \mathcal{T} \rightarrow \mathcal{T}'$ such that $\kappa \circ \eta = \eta'$
 147 and $\kappa \circ \mu = \mu' \circ \kappa\kappa$.

148 A recurrent monad in our work is the powerset monad \mathcal{P}_κ bounded by a cardi-
 149 nal κ . It maps a set X to the set $\mathcal{P}_\kappa X = \{U \mid U \subseteq X, U \text{ has cardinality at most } \kappa\}$
 150 and a function $f: X \rightarrow Y$ to $\mathcal{P}_\kappa f: \mathcal{P}_\kappa X \rightarrow \mathcal{P}_\kappa Y$, $\mathcal{P}_\kappa f(U) = \{f(u) \mid u \in U\}$.
 151 The unit η of \mathcal{P}_κ is given by singleton, i.e., $\eta(x) = \{x\}$ and the multiplication μ
 152 is given by union, i.e., $\mu(S) = \bigcup_{U \in S} U$ for $S \in \mathcal{P}_\kappa \mathcal{P}_\kappa X$.

153 For instance, with $\kappa = \omega$, \mathcal{P}_ω is the *finite* powerset monad, mapping a set
 154 to its finite subsets.

155 **Free monads.** We recall the construction of the monad $\mathcal{F}^\dagger: \mathcal{C} \rightarrow \mathcal{C}$ *freely*
 156 *generated* by a functor $\mathcal{F}: \mathcal{C} \rightarrow \mathcal{C}$. Assume that \mathcal{C} has coproducts and that,
 157 for all objects X of \mathcal{C} , there exists an initial $X + \mathcal{F}$ -algebra that we denote
 158 as $X + \mathcal{F}(\mathcal{F}^\dagger X) \xrightarrow{[\eta_X, \kappa_X]} \mathcal{F}^\dagger X$. It is easy to check that the assignment $X \mapsto$
 159 $\mathcal{F}^\dagger X$ induces a functor $\mathcal{F}^\dagger: \mathcal{C} \rightarrow \mathcal{C}$. The map $\eta_X: X \rightarrow \mathcal{F}^\dagger X$ gives rise to
 160 the unit of the monad; the multiplication $\mu_X: \mathcal{F}^\dagger \mathcal{F}^\dagger X \rightarrow \mathcal{F}^\dagger X$ is the unique
 161 algebra morphism from the initial $\mathcal{F}^\dagger X + \mathcal{F}$ -algebra to the algebra $\mathcal{F}^\dagger X +$
 162 $\mathcal{F}(\mathcal{F}^\dagger X) \xrightarrow{[id, \kappa_X]} \mathcal{F}^\dagger X$.

163 **Distributive laws and bialgebras.** A *distributive law* of a monad (\mathcal{T}, η, μ)
 164 over an endofunctor \mathcal{F} is a natural transformation $\lambda: \mathcal{T}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}$ s.t. $\lambda \circ \eta_{\mathcal{F}} = \mathcal{F}\eta$
 165 and $\lambda \circ \mu_{\mathcal{F}} = \mathcal{F}\mu \circ \lambda_{\mathcal{T}} \circ \mathcal{T}\lambda$. A λ -*bialgebra* is a triple (X, α, β) s.t. (X, α) is an
 166 Eilenberg-Moore algebra for \mathcal{T} , (X, β) is a \mathcal{F} -coalgebra and $\mathcal{F}\alpha \circ \lambda_X \circ \mathcal{T}\beta = \beta \circ \alpha$.
 167 Bialgebra morphisms are both \mathcal{T} -algebra and \mathcal{F} -coalgebra morphisms.

168 Given a coalgebra $\beta: X \rightarrow \mathcal{F}\mathcal{T}X$ for a monad (\mathcal{T}, η, μ) and a functor \mathcal{F} ,
 169 if there exists a distributive law $\lambda: \mathcal{T}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}$, one can form a coalgebra
 170 $\beta^\sharp: \mathcal{T}X \rightarrow \mathcal{F}\mathcal{T}X$ defined as $\mathcal{T}X \xrightarrow{\mathcal{T}\beta} \mathcal{T}\mathcal{F}\mathcal{T}X \xrightarrow{\lambda_{\mathcal{T}X}} \mathcal{F}\mathcal{T}\mathcal{T}X \xrightarrow{\mathcal{F}\mu} \mathcal{F}\mathcal{T}X$. Most
 171 importantly, $(\mathcal{T}X, \mu, \beta^\sharp)$ is a λ -bialgebra.

172 **GSOS specifications.** An *abstract GSOS specification* is a natural trans-
 173 formation $\lambda: \mathcal{S}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}^\dagger$, where \mathcal{F} is a functor representing the coalgebraic
 174 behaviour, \mathcal{S} is a functor representing the syntax. It is important to recall the
 175 following fact.

176 **Proposition 1** ([?]). *Any GSOS spec. $\lambda: \mathcal{S}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}^\dagger$ yields a distrib. law*
 177 $\lambda^\dagger: \mathcal{S}^\dagger\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}^\dagger$.

178 We refer the reader to ?? for a fully developed example of a GSOS specification,
 179 in the context of the semantics of nondeterministic automata.

Coproduct of GSOS specifications. Suppose to have two functors $\mathcal{S}_1, \mathcal{S}_2: \mathcal{C} \rightarrow$
 \mathcal{C} modelling two syntaxes and a functor $\mathcal{F}: \mathcal{C} \rightarrow \mathcal{C}$ modelling the coalgebraic
 behaviour, such that there are two GSOS specifications

$$\lambda_1: \mathcal{S}_1\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_1^\dagger \quad \text{and} \quad \lambda_2: \mathcal{S}_2\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_2^\dagger.$$

Then we can construct a GSOS specification

$$\lambda_1 \cdot \lambda_2: (\mathcal{S}_1 + \mathcal{S}_2)\mathcal{F} \Rightarrow \mathcal{F}(\mathcal{S}_1 + \mathcal{S}_2)^\dagger$$

as follows

$$\begin{array}{ccc}
 \mathcal{S}_1\mathcal{F} & \xrightarrow{\lambda_1} & \mathcal{F}\mathcal{S}_1^\dagger \\
 \gamma_1 \downarrow & & \downarrow \mathcal{F}\iota_1 \\
 (\mathcal{S}_1 + \mathcal{S}_2)\mathcal{F} \cong \mathcal{S}_1\mathcal{F} + \mathcal{S}_2\mathcal{F} & \dashrightarrow & \mathcal{F}(\mathcal{S}_1 + \mathcal{S}_2)^\dagger \\
 \gamma_2 \uparrow & & \uparrow \mathcal{F}\iota_2 \\
 \mathcal{S}_2\mathcal{F} & \xrightarrow{\lambda_2} & \mathcal{F}\mathcal{S}_2^\dagger
 \end{array}$$

180 In the above diagram, the dashed map is given by universal property of the
 181 coproduct $\mathcal{S}_1\mathcal{F} + \mathcal{S}_2\mathcal{F}$. The definitions of γ_1 and γ_2 are as follows. For $X \in \mathcal{C}$,
 182 $\gamma_1(X)$ is the unique $X + \mathcal{S}_1$ -algebra map from the initial $X + \mathcal{S}_1$ -algebra $\mathcal{S}_1^\dagger(X)$
 183 to $(\mathcal{S}_1 + \mathcal{S}_2)^\dagger(X)$. Indeed, since $(\mathcal{S}_1 + \mathcal{S}_2)^\dagger(X)$ is the initial $X + \mathcal{S}_1 + \mathcal{S}_2$ -algebra,
 184 it is in particular a $X + \mathcal{S}_1$ -algebra by precomposition with the coproduct map
 185 $X + \mathcal{S}_1 \rightarrow X + \mathcal{S}_1 + \mathcal{S}_2$. The definition of $\gamma_2(X)$ is analogous, using the fact
 186 that $(\mathcal{S}_1 + \mathcal{S}_2)^\dagger(X)$ is also a $X + \mathcal{S}_2$ -algebra.

187 **Quotients of monads and distributive laws.** Given the correspondence
 188 between finitary monads and algebraic theories [?], it natural to consider
 189 *quotients* of monads by additional equations. Following [? ? ?], for a monad
 190 \mathcal{T} on a category \mathcal{C} , \mathcal{T} -*equations* can be defined as a tuple $\mathbb{E} = (\mathcal{A}, l, r)$ consisting
 191 of a functor $\mathcal{A}: \mathcal{C} \rightarrow \mathcal{C}$ and natural transformations $l, r: \mathcal{A} \Rightarrow \mathcal{T}$. The intuition
 192 is that \mathcal{A} acts as the variables of each equation, whose left- and right-hand
 193 sides are l and r , respectively. Assuming mild conditions that generalise the
 194 properties of \mathbf{Set} (see [? , Ass. 7.1.2]), one constructs the *quotient* of \mathcal{T} by
 195 \mathcal{T} -equations. The conditions hold in our setting: categories of presheaves over
 196 a discrete index category.

197 **Proposition 2** (cf. [?]). If $\mathcal{C} = \mathbf{Set}^{\mathcal{D}}$ for discrete \mathcal{D} , \mathcal{T} -equations \mathbb{E} yield a
 198 monad $\mathcal{T}_{\mathbb{E}}: \mathcal{C} \rightarrow \mathcal{C}$ with algebras precisely \mathcal{T} -algebras $\mathcal{T}A \xrightarrow{\alpha} A$ that satisfy \mathbb{E} ,
 199 in the sense that $\alpha \circ l_A = \alpha \circ r_A$. Moreover, there exists a monad morphism
 200 $q^{\mathbb{E}}: \mathcal{T} \rightarrow \mathcal{T}_{\mathbb{E}}$ with epi components.

One may also quotient distributive laws, provided these are compatible with the new equations. Fix an endofunctor \mathcal{F} and a monad \mathcal{T} on $\mathbf{Set}^{\mathcal{D}}$, together with \mathcal{T} -equations $\mathbb{E} = (\mathcal{A}, l, r)$. We say that a distributive law $\lambda: \mathcal{T}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}$ preserves equations \mathbb{E} if, for all $A \in \mathcal{C}$, the following diagram commutes:

$$\mathcal{A}\mathcal{F}A \xrightarrow[r_{\mathcal{F}A}]{} \mathcal{T}\mathcal{F}A \xrightarrow{\lambda_A} \mathcal{F}\mathcal{T}A \xrightarrow{\mathcal{F}q_A^{\mathbb{E}}} \mathcal{F}\mathcal{T}_{\mathbb{E}}A$$

201 **Proposition 3** (cf. [?]). If $\lambda: \mathcal{T}\mathcal{F} \rightarrow \mathcal{F}\mathcal{T}$ preserves equations \mathbb{E} then there
 202 exists a (unique) distributive law $\lambda_{/\mathbb{E}}: \mathcal{T}_{\mathbb{E}}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}_{\mathbb{E}}$ such that $\lambda_{/\mathbb{E}} \circ q^{\mathbb{E}} = \mathcal{F}q^{\mathbb{E}} \circ \lambda$.

203 4. Diagrammatic Syntax as Monads

204 4.1. The Category of Signatures

205 Syntax and semantics of string diagrams will be specified in the category
 206 $\mathbf{Sig} := \mathbf{Span}(\mathbf{Set})(\mathbb{N}, \mathbb{N})$, where objects are spans $\mathbb{N} \leftarrow \Sigma \rightarrow \mathbb{N}$ in \mathbf{Set} and arrows
 207 are span morphisms: given objects $\mathbb{N} \xleftarrow{s} X \xrightarrow{t} \mathbb{N}$ and $\mathbb{N} \xleftarrow{s'} \Sigma' \xrightarrow{t'} \mathbb{N}$, an arrow is
 208 a function $f: \Sigma \rightarrow \Sigma'$ such that $t' \circ f = t$ and $s' \circ f = s$. We think of an object
 209 of \mathbf{Sig} as a *signature*, i.e. a set of symbols Σ equipped with arity and coarity
 210 functions $a, c: \Sigma \rightarrow \mathbb{N}$. We write $\Sigma(n, m)$ for the set $\{d \in \Sigma \mid \langle a, c \rangle(d) = (n, m)\}$
 211 of operations with arity n and coarity m . Note that we allow coarities different
 212 from 1: this is because string diagrams express *monoidal* algebraic theories, not
 213 merely *cartesian* ones.

214 Since the objects in \mathbf{Sig} are spans with identical domain and codomain, we
 215 will often write Σ for the entire span $\mathbb{N} \xleftarrow{a} \Sigma \xrightarrow{c} \mathbb{N}$. In particular, \mathbb{N} means the
 216 identity span $\mathbb{N} \xleftarrow{id} \mathbb{N} \xrightarrow{id} \mathbb{N}$.

217 **Example 4.** Recall the language \mathbf{Circ}_R from § ???. Line (??)-(??) of its syntax
 218 together with the first two lines of the sorting discipline in Fig. ?? define a
 219 signature Σ : every axiom $d: (n, m)$ gives the symbol d arity n and coarity m .
 220 For instance, $\Sigma(1, 2) = \{-\bullet\overline{\square}, -\circ\overline{\square}\}$.

221 For computing (co)limits, it is useful to observe that \mathbf{Sig} is isomorphic to
 222 the presheaf category $\mathbf{Set}^{\mathbb{N} \times \mathbb{N}}$, where $\mathbb{N} \times \mathbb{N}$ is the discrete category with objects
 223 pairs $(n, m) \in \mathbb{N} \times \mathbb{N}$.

224 4.2. Functors on Signatures

We turn to (co)algebras of endofunctors $\mathcal{F}: \mathbf{Sig} \rightarrow \mathbf{Sig}$ generated by the following grammar:

$$\mathcal{F} ::= Id \mid \Sigma \mid \mathbb{N} \mid \mathcal{F}; \mathcal{F} \mid \mathcal{F} \oplus \mathcal{F} \mid \mathcal{F} + \mathcal{F} \mid \mathcal{F} \times \mathcal{F} \mid \overline{\mathcal{G}}$$

225 where \mathcal{G} ranges over functors $\mathcal{G} : \mathbf{Set} \rightarrow \mathbf{Set}$ and Σ is a span $\mathbb{N} \leftarrow \Sigma \rightarrow \mathbb{N}$. In
 226 more detail:

- 227 • $Id : \mathbf{Sig} \rightarrow \mathbf{Sig}$ is the identity functor.
- 228 • $\Sigma : \mathbf{Sig} \rightarrow \mathbf{Sig}$ is the constant functor mapping every object to $\mathbb{N} \leftarrow \Sigma \rightarrow \mathbb{N}$
 229 and every arrow to id_Σ ; an important special case is $\mathbb{N} : \mathbf{Sig} \rightarrow \mathbf{Sig}$ the
 230 constant functor to $\mathbb{N} \xleftarrow{id} \mathbb{N} \xrightarrow{id} \mathbb{N}$.
- $(\cdot); (\cdot) : \mathbf{Sig}^2 \rightarrow \mathbf{Sig}$ is *sequential composition* for signatures. On objects,
 $\Sigma_1; \Sigma_2$ is

$$\mathbb{N} \xleftarrow{s_1 \circ \pi_1} \{(d_1, d_2) \in \Sigma_1 \times \Sigma_2 \mid t_1(d_1) = s_2(d_2)\} \xrightarrow{t_2 \circ \pi_2} \mathbb{N}.$$

231 Since the above is a **Set**-pullback, the action on arrows is inducted by the
 232 universal property. Note that, up to iso, $(\cdot); (\cdot) : \mathbf{Sig}^2 \rightarrow \mathbf{Sig}$ is associative
 233 with unit $\mathbb{N} : \mathbf{Sig} \rightarrow \mathbf{Sig}$.

- $(\cdot) \oplus (\cdot) : \mathbf{Sig}^2 \rightarrow \mathbf{Sig}$ is *parallel composition* for signatures, with $\Sigma_1 \oplus \Sigma_2$
 given by:

$$\mathbb{N} \xleftarrow{+ \circ (s_1 \times s_2)} \Sigma_1 \times \Sigma_2 \xrightarrow{+ \circ (t_1 \times t_2)} \mathbb{N}$$

234 where $+$: $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ is usual \mathbb{N} -addition. Again $(\cdot) \oplus (\cdot) : \mathbf{Sig}^2 \rightarrow \mathbf{Sig}$
 235 associates up to iso.

- 236 • For the remaining functors, we use the fact that $\mathbf{Sig} \cong \mathbf{Set}^{\mathbb{N} \times \mathbb{N}}$, which guar-
 237 antees (co)completeness, with limits and colimits constructed pointwise in
 238 **Set**. Thus, for spans Σ_1 and Σ_2 , their coproduct is $\mathbb{N} \xleftarrow{[s_1, s_2]} \Sigma_1 + \Sigma_2 \xrightarrow{[t_1, t_2]} \mathbb{N}$
 239 and the carrier of the product is $\{(d_1, d_2) \mid s_1(d_1) = s_2(d_2) \text{ and } t_1(d_1) =$
 240 $t_2(d_2)\}$, with the two obvious morphisms to \mathbb{N} .
- 241 • The isomorphism $\mathbf{Sig} \cong \mathbf{Set}^{\mathbb{N} \times \mathbb{N}}$ also yields the extension of an arbitrary
 242 endofunctor $\mathcal{G} : \mathbf{Set} \rightarrow \mathbf{Set}$ to a functor $\bar{\mathcal{G}} : \mathbf{Sig} \rightarrow \mathbf{Sig}$ defined by post-
 243 composition with \mathcal{G} , that is $\bar{\mathcal{G}}(\Sigma) = \mathcal{G} \circ \Sigma$ for all $\Sigma : \mathbb{N} \times \mathbb{N} \rightarrow \mathbf{Set}$. In
 244 particular, we shall often use the functor $\overline{\mathcal{P}_\kappa}$ obtained by post-composition
 245 with the κ -bounded powerset functor $\mathcal{P}_\kappa : \mathbf{Set} \rightarrow \mathbf{Set}$.¹

246 Next we use these endofunctors to construct monads that capture the two-
 247 dimensional algebraic structure of string diagrams. In § ?? we construct the
 248 monad encoding the symmetric monoidal structure of props and in § ?? we con-
 249 struct the monad for the Frobenius structure of Carboni-Walters props. Later,
 250 in § ??, we shall use these monads to define compositional bialgebraic semantics
 251 for string diagrams of each of these categorical structures.

¹Boundedness is needed to ensure the existence of a final coalgebra, see § ?. In our leading
 example \mathbf{Circ}_R , κ can be taken to be the cardinality of the semiring R .

252 *4.3. The Prop Monad*

Here we define a monad on \mathbf{Sig} with algebras precisely props: symmetric strict monoidal categories with objects the natural numbers, where the monoidal product on objects is addition. Together with identity-on-objects symmetric monoidal functors they form a category **PROP**. The first step is to encapsulate the operations of props as a \mathbf{Sig} -endofunctor.

$$\mathcal{S}_{\mathbf{SM}} := (Id; Id) + \iota + (Id \oplus Id) + \epsilon + \sigma : \mathbf{Sig} \rightarrow \mathbf{Sig}. \quad (11)$$

253 In the type of $\mathcal{S}_{\mathbf{SM}}$, $Id; Id : \mathbf{Sig} \rightarrow \mathbf{Sig}$ is sequential composition and ι the identity
 254 arrow on object 1, i.e. the constant functor to $\mathbb{N} \xleftarrow{h} \{id_1\} \xrightarrow{h} \mathbb{N}$, with $h: id_1 \mapsto 1$.
 255 Similarly, $Id \oplus Id$ is the monoidal product with unit ϵ , i.e. the constant functor
 256 to $\mathbb{N} \xleftarrow{q} \{0\} \xrightarrow{q} \mathbb{N}$, with $q: 0 \mapsto 0$. Finally, σ is the basic symmetry: the constant
 257 functor to $\mathbb{N} \xleftarrow{f} \{\sigma_{1,1}\} \xrightarrow{f} \mathbb{N}$, with $f: \sigma_{1,1} \mapsto 2$.

258 The free monad $\mathcal{S}_{\mathbf{SM}}^\dagger$ on $\mathcal{S}_{\mathbf{SM}}$ is the functor mapping a span Σ to the span of
 259 Σ -terms obtained by sequential and parallel composition, together with symmetries
 260 and identities —with the identity id_n defined by parallel composition of n
 261 copies of id_1 .

Algebras for this monad are spans Σ together with span morphisms *identity*: $\iota \rightarrow \Sigma$, *composition*: $\Sigma; \Sigma \rightarrow \Sigma$, *parallel*: $\Sigma \oplus \Sigma \rightarrow \Sigma$, *unit*: $\epsilon \rightarrow \Sigma$, and *swap*: $\sigma \rightarrow \Sigma$. This information *almost* defines a prop \mathcal{C}_Σ : the carrier Σ of the span is the set of arrows of \mathcal{C}_Σ , containing special arrows id_n and $\sigma_{n,m}$ for identities and symmetries, *compose* assigns to every pairs of composable arrows their composition, and \oplus assigns to every pair of arrows their monoidal product. The missing data is the usual equations (??)-(??) of symmetric monoidal categories. Thus, in order to obtain props as algebras, we quotient the monad $\mathcal{S}_{\mathbf{SM}}^\dagger$ by those equation, expressed abstractly as a triple $\mathbb{E}_{\mathbf{SM}} = (\mathcal{A}, l, r)$, as described in § ???. The functor $\mathcal{A}: \mathbf{Sig} \rightarrow \mathbf{Sig}$, defined below, has summands following the order (??)-(??):

$$\begin{aligned} & (Id \oplus Id \oplus Id) + Id + Id + \sigma + ((Id; Id) \oplus (Id; Id)) \\ & + (Id; Id; Id) + Id + Id + Id^{+1} + Id^{+1} \end{aligned} \quad (12)$$

262 Here, Id^{+1} is the functor adding 1 to the arity/coarity of each element of a
 263 given span $\mathbb{N} \xleftarrow{a} \Sigma \xrightarrow{c} \mathbb{N}$. We also need natural transformations $l, r: \mathcal{A} \rightarrow \mathcal{S}_{\mathbf{SM}}^\dagger$
 264 that define the left- and right-hand side of each equation. For instance, for fixed
 265 $\Sigma \in \mathbf{Sig}$ and $(n, m) \in \mathbb{N} \times \mathbb{N}$:

- 266 • an element of $\Sigma; \Sigma; \Sigma$ (sixth summand of (??)) is a tuple (f, g, h) of Σ -
 267 elements, where f is of type (n, w) , g of type (w, v) , and h of type (v, m) ,
 268 for arbitrary $w, v \in \mathbb{N}$. We let l_Σ map (f, g, h) to the term $(f; g); h$ of
 269 type (n, m) in $\mathcal{S}_{\mathbf{SM}}^\dagger(\Sigma)$, and r_Σ to the term $f; (g; h)$. Thus this component
 270 gives the second equation in (??) (associativity).
- 271 • the seventh summand Id in (??) yields a Σ -term f , which $l_\Sigma: \Sigma \rightarrow \mathcal{S}_{\mathbf{SM}}^\dagger(\Sigma)$
 272 maps to $f; id_m$ and $r_\sigma: \Sigma \rightarrow \mathcal{S}_{\mathbf{SM}}^\dagger(\Sigma)$ maps to f , thus yielding the final
 273 equation in (??).

274 • an element in Σ^{+1} (last summand of (??)) of type $(n + 1, m + 1)$ is a
 275 Σ -term g of type (n, m) , which is mapped by l_Σ to $(\sigma_{n,1}; (id_1 \oplus g))$ and
 276 by r_σ to $(g \oplus id_1); \sigma_{m,1}$, both elements of $\mathcal{S}_{SM}^\dagger(\Sigma)$ of type $(n + 1, m + 1)$,
 277 thus giving the final equation in (??).

278 The remainder of the definition of $l, r: \mathcal{A} \rightarrow \mathcal{S}_{SM}^\dagger$, handles the remaining equa-
 279 tions in (??)-(??), and should be clear from the above. Now, using Proposi-
 280 tion ??, we quotient the monad \mathcal{S}_{SM}^\dagger by (\mathcal{A}, l, r) , obtaining a monad that we call
 281 \mathcal{S}_{PROP} . We can then conclude by construction that the Eilenberg-Moore category
 282 $EM(\mathcal{S}_{PROP})$ for the monad \mathcal{S}_{PROP} (with objects the \mathcal{S}_{PROP} -algebras, and arrows the
 283 \mathcal{S}_{PROP} -algebra homomorphisms) is precisely **PROP**.

284 **Proposition 5.** $EM(\mathcal{S}_{PROP}) \cong \mathbf{PROP}$.

285 **Example 6.** The monad \mathcal{S}_{PROP} takes Σ to the prop freely generated by Σ . Taking
 286 Σ as in Example ??, one obtains $\mathcal{S}_{PROP}(\Sigma)$ with arrows $n \rightarrow m$ string diagrams
 287 of \mathbf{Circ}_R of sort (n, m) .

288 4.4. The Carboni-Walters Monad

289 The treatment we gave to props may be applied to other categorical struc-
 290 tures. For space reasons, we only consider one additional such structure: *Carboni-*
 291 *Walters* (CW) props, also called ‘hypergraph categories’ [?]. Here each ob-
 292 ject n carries a distinguished special Frobenius bimonoid compatible with the
 293 monoidal product: it can be defined recursively using parallel compositions of
 294 the Frobenius structure on the generating object 1.

295 **Definition 7.** A CW prop is a prop with morphisms $\dashv\!\!\!\dashv : 1 \rightarrow 2$, $\dashv : 1 \rightarrow 0$,
 296 $\dashv\!\!\!\dashv : 2 \rightarrow 1$, $\dashv : 0 \rightarrow 1$ satisfying the equations of special Frobenius bimonoids
 297 (Fig. ??).

298 CW props with prop morphisms preserving the Frobenius bimonoid form a
 299 subcategory **CW** of **PROP**. We can now extend the prop monad of § ?? to
 300 obtain a monad with algebras CW props. The signature is that of a prop with
 301 the additional Frobenius structure. Let $\dashv\!\!\!\dashv : \mathbf{Sig} \rightarrow \mathbf{Sig}$ be the functor constant
 302 at $\mathbb{N} \xleftarrow{s} \{\dashv\!\!\!\dashv\} \xrightarrow{t} \mathbb{N}$ with $s(\dashv\!\!\!\dashv) = 1$ and $t(\dashv\!\!\!\dashv) = 2$. Similarly, we introduce
 303 the constant functors $\dashv : \mathbf{Sig} \rightarrow \mathbf{Sig}$, $\dashv\!\!\!\dashv : \mathbf{Sig} \rightarrow \mathbf{Sig}$ and $\dashv : \mathbf{Sig} \rightarrow \mathbf{Sig}$ for
 304 the other generators. Let $\mathcal{S}_{FR} := \mathcal{S}_{PROP} + \dashv\!\!\!\dashv + \dashv + \dashv\!\!\!\dashv + \dashv$.

305 We now need to quotient \mathcal{S}_{FR} by the defining equations of special Frobenius
 306 bimonoids (Fig. ??). We omit the detailed encoding of these equations as a
 307 triple $\mathbb{E}_{CW} = (\mathcal{A}_{CW}, l_{CW}, r_{CW})$ since it presents no conceptual difficulty. Let \mathcal{S}_{CW} be
 308 the quotient of \mathcal{S}_{FR} by these equations. As for props, we obtain $EM(\mathcal{S}_{CW}) \cong \mathbf{CW}$
 309 by construction.

310 5. Bialgebraic Semantics for String Diagrams

Now that we have established monads for our categorical structures of inter-
 est, we study coalgebras that capture behaviour for string diagrams in these

categories, and distributive laws that yield the desired bialgebraic semantics. We fix our ‘behaviour’ functor to

$$\mathcal{F} := \overline{\mathcal{P}_\kappa}(L; Id; L): \mathbf{Sig} \rightarrow \mathbf{Sig}$$

311 where $L: \mathbf{Sig} \rightarrow \mathbf{Sig}$ is the *label functor* constant at the span $\mathbb{N} \xleftarrow{|\cdot|} A^* \xrightarrow{|\cdot|} \mathbb{N}$,
 312 with A^* the set of words on some set of labels A . The map $|\cdot|: A^* \rightarrow \mathbb{N}$
 313 takes $w \in A^*$ to its length $|w| \in \mathbb{N}$. An \mathcal{F} -coalgebra is a span morphism
 314 $\Sigma \rightarrow \overline{\mathcal{P}_\kappa}(L; \Sigma; L)$; a function that takes $f \in \Sigma(n, m)$ to a set of transitions
 315 (v, g, w) with the appropriate sorts, i.e. $g \in \Sigma(n, m)$, $|v| = n$ and $|w| = m$.

The data of an \mathcal{F} -coalgebra $\beta: \Sigma \rightarrow \overline{\mathcal{P}_\kappa}(L; \Sigma; L)$ is that of a transition relation. For instance, fix labels $A = \{a, b\}$ and let $x, y \in \Sigma(1, 2)$ and $z \in \Sigma(1, 1)$; suppose also that β maps x to $\{(b; y; ab), (a; x; aa)\}$, y to \emptyset and z to $\{(b; z; a)\}$. Then β can be written:

$$x \xrightarrow[\frac{b}{ab}]{} y \quad x \xrightarrow[\frac{a}{aa}]{} x \quad z \xrightarrow[\frac{b}{a}]{} z \quad (13)$$

316 **Example 8.** In our main example, Fig. ?? defines a coalgebra $\beta: \Sigma \rightarrow \overline{\mathcal{P}_\kappa}(L; \Sigma; L)$
 317 where Σ is the signature from Example ?? and the set of labels is \mathbf{R} . For in-
 318 stance $\beta(-\bullet) = \{(k, -\bullet, \varepsilon) \mid k \in \mathbf{R}\}$. Note the κ bounding \mathcal{P}_κ is the cardinality
 319 of \mathbf{R} .

320 In the sequel we shall construct distributive laws between the above be-
 321 haviour functor and monads encoding the various categorical structures defined
 322 in the previous section.

323 5.1. Existence of Final Coalgebras

324 First, we prove the existence of the final coalgebra associated to each be-
 325 haviour functor.

326 **Proposition 9.** *There exists a final coalgebra $\Omega \rightarrow \mathcal{F}(\Omega)$.*

327 *Proof.* Being isomorphic to a presheaf category, \mathbf{Sig} is accessible. Then one may
 328 construct Ω via a so-called terminal sequence [?], provided that (i) \mathcal{F} preserves
 329 monomorphisms and (ii) is accessible. Because functor composition preserve
 330 these properties, it suffices to check them componentwise on $\mathcal{F} = \overline{\mathcal{P}_\kappa}(L; Id; L)$.
 331 First, $(-; -)$ is defined by pullbacks, which preserve monomorphisms and (filtered)
 332 colimits; it follows that $(-; -)$ satisfies (i) and (ii). Next, the functor
 333 $\overline{\mathcal{P}_\kappa}$ satisfies (i) for the same reason as the arbitrary powerset functor, and it
 334 satisfies (ii) because it is a κ -bounded functor. Finally, satisfaction of (i) and
 335 (ii) is completely obvious for Id and the constant functor L . \square

336 Note that the proof of Proposition ?? is constructive: following [?], for a
 337 fixed $(n, m) \in \mathbb{N} \times \mathbb{N}$, we can visualise the elements of Ω as κ -branching trees
 338 with edges labelled with pairs of labels $(l_1, l_2) \in L \times L$ with $|l_1| = n, |l_2| = m$.
 339 The unique \mathcal{F} -coalgebra map $[\cdot]_\beta: \mathcal{S}_{\text{PROP}}(\Sigma) \rightarrow \Omega$ sends a Σ -term t to the tree
 340 whose branching describes all the executions from t according to the transition
 341 rules defined by β .

342 *5.2. Bialgebraic Semantics for Props*

343 The modularity of $\mathcal{S}_{\text{PROP}}$ can be exploited to define a distributive law of
 344 the $\mathcal{S}_{\text{PROP}}$ over \mathcal{F} . Recall from § ?? that $\mathcal{S}_{\text{PROP}}$ is a quotient of $\mathcal{S}_{\text{SM}}^\dagger$. We start
 345 by letting $\mathcal{F} = \overline{\mathcal{P}_\kappa}(L; Id; L)$ interact with the individual summands of \mathcal{S}_{SM} (see
 346 (??)), corresponding to the operations of props. This amounts to defining GSOS
 347 specifications:

- sequential composition

$$\lambda^{\text{sq}}: \overline{\mathcal{P}_\kappa}(L; Id; L); \overline{\mathcal{P}_\kappa}(L; Id; L) \Rightarrow \overline{\mathcal{P}_\kappa}(L; (Id; Id)^\dagger; L)$$

- identity

$$\lambda^{\text{id}}: \iota \Rightarrow \overline{\mathcal{P}_\kappa}(L; \iota^\dagger; L)$$

- monoidal product

$$\lambda^{\text{mp}}: \overline{\mathcal{P}_\kappa}(L; Id; L) \oplus \overline{\mathcal{P}_\kappa}(L; Id; L) \Rightarrow \overline{\mathcal{P}_\kappa}(L; (Id \oplus Id)^\dagger; L)$$

- monoidal unit

$$\lambda^\epsilon: \epsilon \Rightarrow \overline{\mathcal{P}_\kappa}(L; \epsilon^\dagger; L)$$

- symmetry

$$\lambda^{\text{sy}}: \sigma \Rightarrow \overline{\mathcal{P}_\kappa}(L; \sigma^\dagger; L)$$

348 Definitions of these maps are succinctly given via derivation rules, in Fig. ??.

We explain this in detail for λ^{sq} , the others are similar. Given $\Sigma \in \text{Sig}$,
 an element of type (n, m) in the domain $\overline{\mathcal{P}_\kappa}(L; \Sigma; L); \overline{\mathcal{P}_\kappa}(L; \Sigma; L)$ is a pair
 (A, B) , where, for some $z \in \mathbb{N}$,

A is a set of triples $(\mathbf{a}, \mathbf{c}', \mathbf{b}) \in L(n, n) \times \Sigma(n, z) \times L(z, z)$, and

B is a set of triples $(\mathbf{b}, \mathbf{d}', \mathbf{c}) \in L(z, z) \times \Sigma(z, m) \times L(m, m)$.

349 Then $\lambda_\Sigma^{\text{sq}}(A, B) := \{(\mathbf{a}, \mathbf{c}'; \mathbf{d}', \mathbf{c}) \mid (\mathbf{a}, \mathbf{c}', \mathbf{b}) \in A, (\mathbf{b}, \mathbf{d}', \mathbf{c}) \in B\}$. Following
 350 the convention (??), we can write this data as: $\boxed{\frac{\mathbf{a}}{\mathbf{c}} \rightarrow \mathbf{c}'; \mathbf{d}'}$ $\in \lambda_\Sigma^{\text{sq}}(A, B)$ if

351 $\boxed{\frac{\mathbf{a}}{\mathbf{b}} \rightarrow \mathbf{c}'}$ $\in A$ and $\boxed{\frac{\mathbf{b}}{\mathbf{c}} \rightarrow \mathbf{d}'}$ $\in B$. This leads us to the more compact version of
 352 λ^{sq} as the transition rule in Fig.??.

353 Next, take the coproduct of GSOS specifications $\lambda^{\text{sq}}, \lambda^{\text{id}}, \lambda^{\text{mp}}, \lambda^\epsilon$ and λ^{sy}
 354 (see [?] for the details) to obtain $\lambda: \mathcal{S}_{\text{SM}}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_{\text{SM}}^\dagger$. By Proposition ??, this
 355 yields distributive law $\lambda^\dagger: \mathcal{S}_{\text{SM}}^\dagger\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_{\text{SM}}^\dagger$.

The last step is to upgrade λ^\dagger to a distributive law $\lambda^\dagger_{/\text{SMC}}$ over the quot-
 tient $\mathcal{S}_{\text{PROP}}$ of $\mathcal{S}_{\text{SM}}^\dagger$ by the equations (??)-(??) of SMCs. By Proposition ??, this
 is well-defined if λ^\dagger preserves \mathbb{E}_{SM} . We show compatibility with associativity
 of sequential composition—the other equations can be verified similarly. This

amounts to checking that if λ^\dagger allows the derivation for $s_1; (s_2; s_3)$ as below left, then there exists a derivation for $(s_1; s_2); s_3$ as on the right, and vice-versa.

$$\frac{s_1 \xrightarrow{u} s'_1 \quad \frac{s_2 \xrightarrow{v} s'_2 \quad s_3 \xrightarrow{w} s'_3}{s_2; s_3 \xrightarrow{x} s'_2; s'_3}}{s_1; (s_2; s_3) \xrightarrow{x} s'_1; (s'_2; s'_3)} \quad \frac{s_1 \xrightarrow{u} s'_1 \quad s_2 \xrightarrow{v} s'_2}{s_1; s_2 \xrightarrow{w} s'_1; s'_2} \quad s_3 \xrightarrow{w} s'_3 \quad (14)}{(s_1; s_2); s_3 \xrightarrow{x} (s'_1; s'_2); s'_3}$$

356 By Proposition ??, we can therefore upgrade λ^\dagger to a distributive law $\lambda^\dagger_{/SM} : \mathcal{S}_{PROP}\mathcal{F} \Rightarrow$
 357 $\mathcal{F}\mathcal{S}_{PROP}$.

358 **Remark 10.** *The above distributive law is not unique: one can devise alterna-*
 359 *tive distributive laws of type $\mathcal{S}_{PROP}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_{PROP}$. Indeed, any structure of prop*
 360 *defined over pairs of labels (as morphisms) would work. We sketch below how*
 361 *to define another distributive law $\mathcal{S}_{CAT}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_{CAT}$, as this is sufficient to make*
 362 *the point.*

First, we can always assume that we have some monoid structure (\cdot, e) for any non-empty finite set of labels A , say by ordering the labels and computing modulo $|A|$. We now show how we can extend this to a monoid (and thus, a category with one object) on elements of A^* . We treat words over A as functions $w : \mathbb{N} \rightarrow A \cup \{\perp\}$ for which there exists some integer $|w|$ (called the length of the word w) such that $w(i) \neq \perp$ for $i \leq |w|$ and $w(i) = \perp$ for $i > |w|$. Then we can extend the binary operation \cdot naturally to $A \cup \{\perp\}$ by $a \cdot \perp = \perp \cdot a = a$. Finally, we extend this further to an associative binary operation on words by computing \cdot pointwise and truncating to the length of the first word:

$$(v \cdot w)(i) = \begin{cases} v(i) \cdot w(i) & \text{if } i \leq |v|, \\ \perp & \text{otherwise.} \end{cases}$$

363 By definition, notice that $|v \cdot w| = |v|$.

Now, let the sequential composition component of the distributive law be given by

$$\frac{s_1 \xrightarrow{v_1} s'_1 \quad s_2 \xrightarrow{v_2} s'_2}{s_1; s_2 \xrightarrow{v_1 \cdot v_2} s'_1; s'_2}$$

Note the inversion for $w_2 \cdot w_1$, which is needed because we require $|w_2 \cdot w_1| = |w_2|$ for this to be a distributive law. This definition guarantees that we can replicate the two derivations from (??), where the labels of the last transitions are the same.

$$\frac{s_1 \xrightarrow{v_1} s'_1 \quad \frac{s_2 \xrightarrow{v_2} s'_2 \quad s_3 \xrightarrow{v_3} s'_3}{s_2; s_3 \xrightarrow{v_2 \cdot v_3} s'_2; s'_3}}{s_1; (s_2; s_3) \xrightarrow{v_1 \cdot v_2 \cdot v_3} s'_1; (s'_2; s'_3)} \quad \frac{s_1 \xrightarrow{v_1} s'_1 \quad s_2 \xrightarrow{v_2} s'_2}{s_1; s_2 \xrightarrow{v_1 \cdot v_2} s'_1; s'_2} \quad s_3 \xrightarrow{v_3} s'_3}{(s_1; s_2); s_3 \xrightarrow{v_1 \cdot v_2 \cdot v_3} (s'_1; s'_2); s'_3}$$

The identity component of the distributive law is given by

$$\overline{id_n \xrightarrow{e^n} id_n}$$

where the e^n label is the word of length n containing only e , the unit of the monoid structure on A , e.g., $id_3 \xrightarrow[e^e]{ee} id_3$ for the identity on 3 wires. As for associativity above, we can prove (right) unitality:

$$\frac{s \xrightarrow[w]{v} s' \quad id_n \xrightarrow[e^n]{e^n} id_n}{s; id_n \xrightarrow[e^n \cdot w]{v \cdot e^n} s'; id_n}$$

364 This defines a suitable distributive law $\mathcal{S}_{CAT}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_{CAT}$ because, by definition,
 365 $v \cdot e^n = v$ and $e^n \cdot w = w$ since $|w| = n$.

366 We are now ready to construct the compositional semantics as a morphism
 367 into the final coalgebra. One starts with a coalgebra $\beta: \Sigma \rightarrow \mathcal{F}(\mathcal{S}_{PROP}(\Sigma))$ that
 368 describes the behaviour of Σ -operations, assigning to each a set of transitions,
 369 as in (??). The difference with (??) is that, because \mathcal{F} is applied to $\mathcal{S}_{PROP}(\Sigma)$
 370 instead of just Σ , the right-hand side of each transition contains not just a
 371 Σ -operation, but a *string diagram*: a Σ -term modulo the laws of SMCs.

372 As recalled in § ??, using the distributive law $\lambda^\dagger_{/SM}$ we can lift $\beta: \Sigma \rightarrow$
 373 $\mathcal{F}(\mathcal{S}_{PROP}(\Sigma))$ to a $\lambda^\dagger_{/SM}$ -bialgebra, $\beta^\sharp: \mathcal{S}_{PROP}(\Sigma) \rightarrow \mathcal{F}(\mathcal{S}_{PROP}(\Sigma))$. Since this is a
 374 \mathcal{F} -coalgebra, the final \mathcal{F} -coalgebra Ω (the existence of which is shown in [?])
 375 yields a semantics $\llbracket \cdot \rrbracket_\beta$ as below. The operational semantics of a string diagram
 376 c is $\beta^\sharp(c)$, obtained from (i) transitions for Σ -operations given by β and (ii) the
 377 derivation rules (Fig. ??) of $\lambda^\dagger_{/SM}$. Instead, $\llbracket c \rrbracket_\beta$ is the observable behaviour:
 378 intuitively, its transition systems modulo bisimilarity.

$$\begin{array}{ccc} \mathcal{S}_{PROP}(\Sigma) & \xrightarrow{\llbracket \cdot \rrbracket_\beta} & \Omega \\ \downarrow \beta^\sharp & & \downarrow \\ \mathcal{F}(\mathcal{S}_{PROP}(\Sigma)) & \xrightarrow{\mathcal{F}(\llbracket \cdot \rrbracket_\beta)} & \mathcal{F}(\Omega) \end{array}$$

379 The bialgebraic semantics framework ensures that $\mathcal{S}_{PROP}(\Sigma)$ and Ω are \mathcal{S}_{PROP} -
 380 algebras, which by Proposition ?? are props. This means that the final coalgebra
 381 Ω is a prop and that $\llbracket \cdot \rrbracket_\beta$ is a prop morphism, preserving identities, symmetries
 382 and guaranteeing compositionality: $\llbracket s; t \rrbracket_\beta = \llbracket s \rrbracket_\beta; \llbracket t \rrbracket_\beta$ and $\llbracket s \oplus t \rrbracket_\beta = \llbracket s \rrbracket_\beta \oplus$
 383 $\llbracket t \rrbracket_\beta$.

384 **Example 11.** Coming back to our running example, in Example ?? we showed
 385 that rules in Fig. ?? induce a coalgebra of type $\Sigma \rightarrow \mathcal{F}(\Sigma)$. Since each operation
 386 in Σ is itself a string diagram (formally, via the unit $\eta_\Sigma: \Sigma \rightarrow \mathcal{S}_{PROP}(\Sigma)$), the
 387 same rules induce a coalgebra $\beta: \Sigma \rightarrow \mathcal{F}\mathcal{S}_{PROP}(\Sigma)$, which has the type required for
 388 the above construction. The resulting coalgebra $\beta^\sharp: \mathcal{S}_{PROP}(\Sigma) \rightarrow \mathcal{F}\mathcal{S}_{PROP}(\Sigma)$ as-
 389 signs to each diagram of $\mathcal{S}_{PROP}(\Sigma)$ the set of transitions specified by the combined
 390 operational semantics of Figs. ?? and ??. The preceding discussion implies that,
 391 when e.g. $\mathbf{R} = \mathbb{N}$, bisimilarity for the Petri nets of [?] is a congruence.

392 *5.3. Bialgebraic Semantics for Carboni-Walters Props*

393 In this section we shall see two different ways of extending the GSOS spec-
 394 ification of § ?? for CW props (see § ??). They correspond to the operational
 395 semantics of the black and white (co)monoids as given in Fig. ?. In the next
 396 section, we will see that these two different extensions give rise to two classic
 397 forms of synchronisation: à la Hoare and à la Milner.

398 **Black distributive law.** The first interprets the operations of the Frobenius
 399 structure as label synchronisation: from the black node derivations on the left
 400 of Fig. ? we get GSOS specifications given by natural transformations $-\bullet\curvearrowright \Rightarrow$
 401 $\mathcal{F}(-\bullet\curvearrowright^\dagger)$, $-\bullet \Rightarrow \mathcal{F}(-\bullet^\dagger)$, $\curvearrowright\bullet \Rightarrow \mathcal{F}(\curvearrowright\bullet^\dagger)$, and $\bullet \Rightarrow \mathcal{F}(\bullet^\dagger)$. Recall that,
 402 here, we use the diagrams to denote their associated functors $\mathbf{Sig} \rightarrow \mathbf{Sig}$. By
 403 taking the coproduct of these and λ , the GSOS specification for props from § ??,
 404 we obtain a specification λ_\bullet for \mathcal{S}_{FR} . It is straightforward to verify that $\lambda_\bullet^\dagger : \mathcal{S}_{\text{FR}}^\dagger \mathcal{F} \Rightarrow \mathcal{F} \mathcal{S}_{\text{FR}}^\dagger$ preserves the equations of special Frobenius bimonoids (Fig. ??),
 405 yielding a distributive law $\lambda_{\bullet/\text{CW}}^\dagger : \mathcal{S}_{\text{CW}}^\dagger \mathcal{F} \Rightarrow \mathcal{F} \mathcal{S}_{\text{CW}}^\dagger$. As before, with $\lambda_{\bullet/\text{CW}}^\dagger$ we
 406 obtain a bialgebra $\beta_\bullet^\sharp : \mathcal{S}_{\text{CW}}(\Sigma) \rightarrow \mathcal{F} \mathcal{S}_{\text{CW}}(\Sigma)$ from any coalgebra $\beta : \Sigma \rightarrow \mathcal{F} \mathcal{S}_{\text{CW}}(\Sigma)$.

408 **White distributive law.** When the set of labels A is an *Abelian group*, it
 409 is possible to give a different GSOS specifications for the Frobenius structure,
 410 capturing the group operation of A : from the white node derivations on the
 411 right of Fig. ? we get GSOS specifications $-\curvearrowleft \Rightarrow \mathcal{F}(-\curvearrowleft^\dagger)$, $-\circ \Rightarrow \mathcal{F}(-\circ^\dagger)$,
 412 $\curvearrowleft\circ \Rightarrow \mathcal{F}(\curvearrowleft\circ^\dagger)$, and $\circ \Rightarrow \mathcal{F}(\circ^\dagger)$. Using a now familiar procedure we obtain
 413 a GSOS specifications λ_\circ for \mathcal{S}_{FR} . The group structure on A guarantees [?]
 414 that $\lambda_\circ^\dagger : \mathcal{S}_{\text{FR}}^\dagger \mathcal{F} \Rightarrow \mathcal{F} \mathcal{S}_{\text{FR}}^\dagger$ preserves the equations of special Frobenius bimonoids
 415 (Fig. ??). Therefore we get a distributive law $\lambda_{\circ/\text{CW}}^\dagger : \mathcal{S}_{\text{CW}}^\dagger \mathcal{F} \Rightarrow \mathcal{F} \mathcal{S}_{\text{CW}}^\dagger$.

416 *5.4. Failure of Compositionality for Lawvere Theories*

417 Given the results in this section, one could ask if bialgebraic semantics works
 418 for *any* categorical structure. A notable case in which it fails is that of Lawvere
 419 theories [?]. These can be seen as props with a *natural* comonoid structure on
 420 each object [?]. One may define a monad for Lawvere theories following the
 421 same recipe as above. However, it turns out that this monad is incompatible
 422 with the GSOS specification for the comonoid given in Fig. ?. The problem
 423 comes from the incompatibility of this structure with the nondeterminism of the
 424 behaviour functor. We explain this in more detail below.

The signature of a Lawvere theory is that of a prop with the additional
 comonoids. As for the comonoid part of Frobenius bimonoids (Fig.??), we only
 need to introduce this comonoid on the generating object 1 since all the others
 can be obtained by parallel and sequential composition of these basic morphisms
 and the symmetries. Let $-\curvearrowleft : \mathbf{Sig} \rightarrow \mathbf{Sig}$ and $-\bullet : \mathbf{Sig} \rightarrow \mathbf{Sig}$ be the constant
 functors defined exactly as in § ?. \mathcal{S}_{L} is then obtained as the quotient of
 $\mathcal{S}_{\text{PROP}} + -\curvearrowleft + -\bullet$ by the defining equations of cocommutative comonoids (first
 line of Fig. ??), and the requirement that the comonoids be natural in the sense

that every morphism $f : m \rightarrow n$ should be a comonoid homomorphism:

$$m \boxed{f} \bullet \begin{array}{c} \curvearrowright \\ n \\ \curvearrowleft \end{array} \approx m \bullet \begin{array}{c} \boxed{f} \\ \curvearrowright \\ \boxed{f} \\ \curvearrowleft \end{array} \begin{array}{c} n \\ n \end{array} \quad m \boxed{f} \bullet \approx m \bullet \quad (15)$$

425 The intuitive interpretation is that every morphism can be copied and deleted
426 using the comonoid structure.

427 To give a GSOS specification for \mathcal{S}_L it is natural to extend the specification
428 for props with specifications for the comonoids. These are entirely analogous
429 to the black interpretation of the comonoid part of CW props as copying and
430 deleting operations as given in Fig. ???. However, this specification turns out to
431 be incompatible with the required naturality of comonoids. To see this, consider
432 the following counter-example.

Let Σ be the signature with a single symbol $d : 0 \rightarrow 1$. We specify its
behaviour as the coalgebra $\beta : \Sigma \rightarrow \mathcal{F}\mathcal{S}_L\Sigma$ given by the two transitions

$$d \xrightarrow{a} d \quad d \xrightarrow{b} d$$

Note the use of nondeterminism here: it is this form of nondeterminism combined
with the naturality requirement of (??) that will lead to a contradiction.
The derivation rules of § ?? give us only two possible transitions for d ; $\bullet \curvearrowright$:

$$\frac{\boxed{d} \xrightarrow{a} \boxed{d} \quad \bullet \curvearrowright \xrightarrow{a} \bullet \curvearrowright}{\boxed{d} \bullet \curvearrowright \xrightarrow{a} \boxed{d} \bullet \curvearrowright} \lambda^{\text{sq}}$$

and

$$\frac{\boxed{d} \xrightarrow{b} \boxed{d} \quad \bullet \curvearrowleft \xrightarrow{b} \bullet \curvearrowleft}{\boxed{d} \bullet \curvearrowleft \xrightarrow{b} \boxed{d} \bullet \curvearrowleft} \lambda^{\text{sq}}$$

whereas $f \oplus f$ can also perform the following two transitions:

$$\frac{\boxed{d} \xrightarrow{a} \boxed{d} \quad \boxed{d} \xrightarrow{b} \boxed{d}}{\boxed{d} \xrightarrow{a} \boxed{d} \quad \boxed{d} \xrightarrow{b} \boxed{d}} \lambda^{\text{mp}}$$

and

$$\frac{\boxed{d} \xrightarrow{b} \boxed{d} \quad \boxed{d} \xrightarrow{a} \boxed{d}}{\boxed{d} \xrightarrow{b} \boxed{d} \quad \boxed{d} \xrightarrow{a} \boxed{d}} \lambda^{\text{mp}}$$

This is in contradiction with the leftmost equation of (??) which requires that

$$\boxed{d} \bullet \curvearrowright \approx \begin{array}{c} \boxed{d} \\ \boxed{d} \end{array}$$

433 . Thus the specification would not be compositional.

434 6. Black and White Frobenius as Hoare and Milner Synchronisation

435 The role of this section is twofold: on the one hand we demonstrate how
 436 classical process calculus syntax benefits from a string diagrammatic treatment;
 437 on the other we draw attention towards a surprising observation, namely that
 438 the black and white Frobenius structures discussed previously provide the syn-
 439 chronisation mechanism of, respectively, CSP and CCS.

440 6.1. Syntax

441 We consider a minimal process calculus for simplicity. Assume a countable
 442 set \mathcal{N} of *names*, a_1, a_2, \dots and a set \mathcal{V} of *process variables*, $\mathbf{f}, \mathbf{g}, \dots$, equipped
 443 with a function $ar: \mathcal{V} \rightarrow \mathbb{N}$ that assigns the set of names that the process may
 444 use: $ar(\mathbf{f}) = n$ means that the process \mathbf{f} uses only names $\{a_1, \dots, a_n\}$. This is
 445 Hoare's [?] notion of *alphabet* for process variables.

446 Roughly speaking, in a string diagram, dangling wires perform the job of
 447 variables. To ease the translation of terms to diagrams, we include permutations
 448 of names in the syntax, hereafter denoted by σ . For a permutation $\sigma: \mathcal{N} \rightarrow \mathcal{N}$,
 449 its support is the set $supp(\sigma) = \{a_i \mid a_i \neq \sigma(a_i)\}$; σ is *finitely supported* if
 450 $supp(\sigma)$ is finite. For each finitely supported permutation σ its *degree* is defined
 451 as the greatest $i \in \mathbb{N}$ such that $a_i \in supp(\sigma)$.

The set of processes is defined recursively as follows

$$P := P|P, \nu a_i(P), \mathbf{f}, P\sigma$$

where $a_i \in \mathcal{N}$, $\mathbf{f} \in \mathcal{V}$ and σ is a finitely supported permutation of names. The symbol $|$ stands for the parallel composition of processes. The symbol νa_i stands for the restriction, or hiding, of the name a_i . Observe that there are no primitives for prefixes, non-deterministic choice or recursion: these will appear in the declaration of process variables which we will describe in § ???. The idea here is to separate the behaviour, specified in the declaration of process variables, and the communication topology of the network, given by the syntax above. The notion of alphabet can be defined for all processes as follows:

$$\begin{aligned} al(P|Q) &= al(P) \cup al(Q) \\ al(\nu a_i(P)) &= al(P) \setminus \{a_i\} \\ al(\mathbf{f}) &= \{a_1, \dots, a_{ar(\mathbf{f})}\} \\ al(P\sigma) &= \sigma[al(P)] \end{aligned}$$

From one-dimensional to two-dimensional syntax. We use a typing dis-

cipline to guide the translation of terms to string diagrams:

$$\begin{array}{c}
\frac{n \vdash P \quad n \vdash Q}{n \vdash P|Q} \quad \frac{n+1 \vdash P}{n \vdash \nu a_{n+1}(P)} \quad \frac{ar(\mathbf{f}) = n}{n \vdash \mathbf{f}} \\
(16) \\
\frac{n \vdash P \quad degree(\sigma) \leq n}{n \vdash P\sigma} \quad \frac{n \vdash P}{n+1 \vdash P}
\end{array}$$

452 The meaning of the types is explained by the following lemma, easily proven
453 by induction.

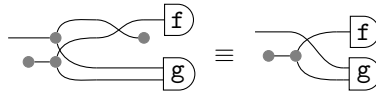
454 **Lemma 12.** *If $n \vdash P$ then $al(P) \subseteq \{a_1, \dots, a_n\}$.*

We will translate processes to the CW prop freely generated from $\Sigma = \{\mathbf{f}: (n, 0) \mid \mathbf{f} \in \mathcal{V} \text{ and } ar(\mathbf{f}) = n\}$; in particular a typed process $n \vdash P$ results in a string diagram of $\mathcal{S}_{CW}(\Sigma)(n, 0)$. The translation $\langle\langle \cdot \rangle\rangle$ is defined recursively on typed terms as follows:

$$\begin{array}{c}
\langle\langle n \vdash P|Q \rangle\rangle = \begin{array}{c} \text{---}^n \bullet \\ \swarrow \quad \searrow \\ \langle\langle P \rangle\rangle \\ \langle\langle Q \rangle\rangle \end{array} \quad \langle\langle n \vdash \nu a_{n+1}(P) \rangle\rangle = \begin{array}{c} \text{---}^n \bullet \\ \searrow \\ \langle\langle P \rangle\rangle \end{array} \\
\langle\langle n \vdash \mathbf{f} \rangle\rangle = \text{---}^n \boxed{\mathbf{f}} \quad \langle\langle n \vdash P\sigma \rangle\rangle = \begin{array}{c} \text{---}^n \boxed{\bar{\sigma}} \text{---}^n \\ \searrow \\ \langle\langle P \rangle\rangle \end{array} \\
\langle\langle n+1 \vdash P \rangle\rangle = \begin{array}{c} \text{---}^n \bullet \\ \searrow \\ \langle\langle P \rangle\rangle \end{array}
\end{array}$$

455 where for σ with $degree(\sigma) < n$, $\bar{\sigma}: n \rightarrow n$ is the obvious corresponding arrow
456 in $\mathcal{S}_{CW}(\Sigma)$.

Example 13. *Let $\mathcal{V} = \{\mathbf{f}, \mathbf{g}\}$ with $ar(\mathbf{f}) = 1$ and $ar(\mathbf{g}) = 2$. Let $[a_2/a_1]: \mathcal{N} \rightarrow \mathcal{N}$ be the permutation swapping a_1 and a_2 . One can easily check that $1 \vdash \nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g})$. Then $\langle\langle 1 \vdash \nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g}) \rangle\rangle$ is as below.*



457 6.2. Semantics

458 In order to give semantics to the calculus, we assume a set \mathcal{A} of actions, $\alpha,$
459 β, \dots . Since, we will consider different sets of actions (for Hoare and Milner
460 synchronisation), we assume them to be functions of type $\mathcal{N} \rightarrow M$ for some
461 monoid $(M, +, 0)$. The support of an action α is the set $\{a_i \mid \alpha(a_i) \neq 0\}$. The
462 alphabet of α , written $al(\alpha)$ is identified with its support.

463 For Hoare synchronisation, the monoid M is $(2, \cup, 0)$, while for Milner it is
464 $(\mathbb{Z}, +, 0)$. In both cases, we will write a_i for the function mapping the name

465 a_i to 1 and all the others to 0. For Milner synchronisation, write \bar{a}_i for the
 466 function mapping a_i to -1 .

To give semantics to processes, we need a *process declaration* for each $\mathbf{f} \in \mathcal{V}$.
 That is, an expression $\mathbf{f} := \sum_{i \in I} \alpha_i.P_i$, for some finite set I , $\alpha_i \in \mathcal{A}$ and processes
 P_i such that

$$\{a_1, \dots, a_{ar(\mathbf{f})}\} \subseteq \bigcup_{i \in I} al(\alpha_i) \cup \bigcup_{i \in I} al(P_i) \quad (17)$$

The basic behaviour of process declarations is captured by the three rules below.

$$\frac{}{\mathbf{f} \xrightarrow{0} \mathbf{f}} \quad \frac{\mathbf{f} := \sum_{i \in I} \alpha_i.P_i}{\mathbf{f} \xrightarrow{\alpha_i} P_i} \quad \frac{P \xrightarrow{\alpha} P'}{P\sigma \xrightarrow{\alpha \circ \sigma} P'\sigma} \quad (18)$$

Example 14. Recall \mathbf{f} and \mathbf{g} from Example ???. Assume the following declarations:

$$\mathbf{f} := a_1.\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g}) \quad \text{and} \quad \mathbf{g} := a_1.\mathbf{g} + a_2.\mathbf{g}.$$

467 Observe that they respect (??). We have that $\mathbf{g} \xrightarrow{a_1} \mathbf{g}$ and $\mathbf{g} \xrightarrow{a_2} \mathbf{g}$ while $\mathbf{f} \xrightarrow{a_1}$
 468 $\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g})$. Similarly $\mathbf{f}[a_2/a_1] \xrightarrow{a_2} (\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g}))[a_2/a_1]$.

469 To define the semantics of parallel and restriction, we need to distinguish
 470 between the Hoare and Milner synchronisation patterns.

Hoare synchronisation. Here actions are functions $\alpha: \mathcal{N} \rightarrow 2$, which can
 equivalently be thought of as subsets of \mathcal{N} . The synchronisation mechanism
 presented below is analogous to the one used in CSP [?]. The main difference
 is the level of concurrency: the classical semantics [?] is purely interleaving,
 while for us it is a step semantics. Essentially, in $P \mid Q$, the processes P and Q
 may evolve independently on the non-shared names, i.e. the evolution of two
 or more processes may happen at the same time. It is for this reason that our
 actions are sets of names. The operational semantics of parallel and restriction
 is given by rules

$$\frac{\frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\beta} Q' \quad \alpha \cap al(Q) = \beta \cap al(P)}{P \mid Q \xrightarrow{\alpha \cup \beta} P' \mid Q'} \quad \frac{}{P \xrightarrow{\alpha} P'}}{\nu a_i(P) \xrightarrow{\alpha \setminus \{a_i\}} \nu a_i(P')} \quad (19)$$

471 We write $\xrightarrow{\alpha}_H$ for the transition systems generated by the rules (??), (??). By a
 472 simple inductive argument, using (??) as base case, we see that for all processes
 473 P , if $P \xrightarrow{\alpha} P'$ then $\alpha \subseteq al(P)$. The rule for parallel, therefore, ensures that P
 474 and Q synchronise over all of their shared names. The rule for restriction hides
 475 a_i from the environment. For instance, if $\alpha = \{a_i\}$, then $\nu a_i(P) \xrightarrow{\emptyset} \nu a_i(P')$. If
 476 $\alpha = \{a_j\}$ with $a_j \neq a_i$, then $\nu a_i(P) \xrightarrow{\{a_j\}} \nu a_i(P')$.

477 **Example 15.** Recall \mathbf{f} and \mathbf{g} from Example ???. We have that $\mathbf{f} \xrightarrow{a_1}_H \nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g})$.
478 From $\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g})$, there are two possibilities: either $\mathbf{f}[a_2/a_1]$ and \mathbf{g} synchronise
479 on a_2 , and in this case we have $\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g}) \xrightarrow{0}$, or \mathbf{g} proceeds without
480 synchronising on a_1 , therefore $\nu a_2(\mathbf{f}[a_2/a_1] \mid \mathbf{g}) \xrightarrow{\{a_1\}}_H$ since a_1 belongs to $al(\mathbf{g})$
481 and not to $al(\mathbf{f}[a_2/a_1])$.

Milner synchronisation. We take $\mathcal{A} = \mathbb{Z}^{\mathcal{N}}$. Sum of functions, denoted by $+$, is defined pointwise and we write 0 for its unit, the constant 0 function.

$$\frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\beta} Q'}{P|Q \xrightarrow{\alpha+\beta} P'|Q'} \quad \frac{P \xrightarrow{\alpha} P'}{\nu a_i(P) \xrightarrow{\alpha} \nu a_i(P')} \alpha(a_i) = 0 \quad (20)$$

482 We write $\xrightarrow{\alpha}_M$ for the transition system generated by the rules (??), (??).

483 Functions in $\mathbb{Z}^{\mathcal{N}}$ to represent concurrent occurrences of CCS send and receive
484 actions. A single CCS action a is the function mapping a to 1 and all other
485 names to 0 . Similarly, the action \bar{a} maps a to -1 and the other names to
486 0 . The silent action τ is the function 0 . With this in mind, it is easy to see
487 that, similarly to CCS, the rightmost rule forbids $\nu a_i(P) \xrightarrow{\alpha} \nu a_i(P')$ whenever
488 $\alpha = a_i$ or $\alpha = \bar{a}_i$. CCS-like synchronisation is obtained by the leftmost rule:
489 when $\alpha = a_i$ and $\beta = \bar{a}_i$, one has that $P|Q \xrightarrow{0} P'|Q'$.

490 A simple inductive argument confirms that $P \xrightarrow{0} P$ for any process P . Then,
491 by the leftmost rule again, one has that whenever $Q \xrightarrow{\beta} Q'$, then $P|Q \xrightarrow{\beta} P'|Q'$.
492 Note, however, that as in § ??, while our synchronisation mechanism is
493 essentially Milner's CSS handshake, our semantics is not interleaving and allows
494 for step concurrency. It is worth remarking that the operational rules in (??)
495 have already been studied by Milner in its work on SCCS [?].

Semantic correspondence. For an action $\alpha: \mathcal{N} \rightarrow M$ with $al(\alpha) \subseteq \{a_1, \dots, a_n\}$,
we write $n \vdash \alpha$ for the restriction $\{a_1, \dots, a_n\} \rightarrow M$. Define coalgebras
 $\beta_b, \beta_w: \Sigma \rightarrow \overline{\mathcal{P}}_{\kappa}(L; \mathcal{S}_{\text{CW}}(\Sigma); L)$ for each $\mathbf{f} \in \Sigma_{n,0}$ where $\mathbf{f} := \sum_{i \in I} \alpha_i \cdot P_i$ as

$$\beta_b(\mathbf{f}) = \beta_w(\mathbf{f}) = \{(n \vdash \alpha_i, \langle\langle P_i \rangle\rangle, \bullet) \mid i \in I\} \cup \{(n \vdash 0, \mathbf{f}, \bullet)\}.$$

496 For both β_b and β_w , L is the span $\mathbb{N} \xleftarrow{|\cdot|} A^* \xrightarrow{|\cdot|} \mathbb{N}$, but $A = 2$ for β_b and $A = \mathbb{Z}$
497 for β_w .

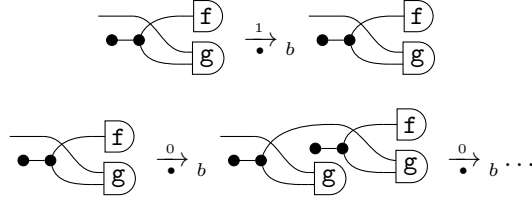
498 Via the distributive law (§ ??) for the black Frobenius, we obtain the coalgebra
499 $\beta_b^{\sharp}: \mathcal{S}_{\text{CW}}(\Sigma) \rightarrow \overline{\mathcal{P}}_{\kappa}(L; \mathcal{S}_{\text{CW}}(\Sigma); L)$. Via the white Frobenius, we obtain
500 $\beta_w^{\sharp}: \mathcal{S}_{\text{CW}}(\Sigma) \rightarrow \overline{\mathcal{P}}_{\kappa}(L; \mathcal{S}_{\text{CW}}(\Sigma); L)$. We write $c \xrightarrow{\beta}_b d$ for $(\alpha, \beta, d) \in \beta_b^{\sharp}(c)$ and
501 $c \xrightarrow{\beta}_w d$ for $(\alpha, \beta, d) \in \beta_w^{\sharp}(c)$. The correspondence can now be stated formally.

502 **Theorem 16.** Let $n \vdash P$ and $n \vdash \alpha$ such that $al(\alpha) \subseteq al(P)$.

503 • **Hoare is black.** If $P \xrightarrow{\alpha}_H P'$ then $n \text{---} \langle\langle P \rangle\rangle \xrightarrow{n \vdash \alpha} n \text{---} \langle\langle P' \rangle\rangle$. Vice
504 versa, if $n \text{---} \langle\langle P \rangle\rangle \xrightarrow{n \vdash \alpha} n \text{---} \langle d \rangle$ then there is $n \vdash P'$ s.t. $P \xrightarrow{\alpha}_H P'$ and
505 $n \text{---} \langle\langle P' \rangle\rangle = n \text{---} \langle d \rangle$.

- 506 • **Milner is white.** If $P \xrightarrow{\alpha}_M P'$ then $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} w n \text{---} \langle\langle P' \rangle\rangle$. Vice
 507 versa, if $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} w n \text{---} \langle d \rangle$ then there is $n \vdash P'$ s.t. $P \xrightarrow{\alpha}_M P'$
 508 and $n \text{---} \langle\langle P' \rangle\rangle = n \text{---} \langle d \rangle$.

Example 17. We illustrate the semantic correspondence by returning to Example ???. Diagrammatically, it yields the following transitions:



509 **6.3. Proof of Theorem ???**

510 We split Theorem ??? into two propositions: Proposition ???, containing the
 511 first part of each statement and Proposition ???, the converse correspondence.
 512 We prove each of these separately below.

513 **Proposition 18.** Let $n \vdash P$.

514 1. If $P \xrightarrow{\alpha}_H P'$ then $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} b n \text{---} \langle\langle P' \rangle\rangle$

515 2. If $P \xrightarrow{\alpha}_M P'$ then $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} w n \text{---} \langle\langle P' \rangle\rangle$

516 *Proof.* We prove 1; the proof for 2 is analogous. Hereafter, given $n \vdash \alpha$, we
 517 write $n + n \vdash \alpha \oplus \alpha$ for the function mapping a_i to $\alpha(a_i)$ if $i \leq n$ and to $\alpha(a_{i-n})$
 518 if $i > n$.

Suppose that $P \xrightarrow{\alpha}_H P'$. We prove by induction on the rules of (??) that

$$n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} b n \text{---} \langle\langle P' \rangle\rangle .$$

- For the rule

$$\frac{n \vdash P \quad n \vdash Q}{n \vdash P|Q}$$

we observe that by the leftmost rule in (??), $(P|Q) \xrightarrow{\gamma}_H R$ iff $R = P'|Q'$,
 $P \xrightarrow{\alpha}_H P'$ and $Q \xrightarrow{\beta}_H Q'$ with $\gamma = \alpha \cup \beta$ and $\alpha \cap al(Q) = \beta \cap al(P)$. By
 the induction hypothesis we have that $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} b n \text{---} \langle\langle P' \rangle\rangle$ and
 $n \text{---} \langle\langle Q \rangle\rangle \xrightarrow[n+\beta]{\bullet} b n \text{---} \langle\langle Q' \rangle\rangle$. Since $\alpha \cap al(Q) = \beta \cap al(P)$ then $(\beta \setminus \alpha) \cap$
 $al(P) = \emptyset$: by Lemma ???, we have that $n \text{---} \langle\langle P \rangle\rangle \xrightarrow[n+\gamma]{\bullet} b n \text{---} \langle\langle P' \rangle\rangle$. By

a symmetric argument, $n \dashv \langle \langle Q \rangle \rangle \xrightarrow[n \bullet]{n \vdash \gamma} b \dashv \langle \langle Q' \rangle \rangle$. Now, $\langle \langle n \vdash P \mid Q \rangle \rangle$

is equal by definition to $n \dashv \left(\begin{array}{c} \langle \langle P \rangle \rangle \\ \langle \langle Q \rangle \rangle \end{array} \right)$. Since $n \dashv \left(\begin{array}{c} n \\ n \end{array} \xrightarrow[n \bullet]{n \vdash \gamma} \begin{array}{c} n \\ n + n \vdash \gamma \oplus \gamma \end{array} \xrightarrow[n \bullet]{n} n \right)$, then

$$n \dashv \left(\begin{array}{c} \langle \langle P \rangle \rangle \\ \langle \langle Q \rangle \rangle \end{array} \right) \xrightarrow[n \bullet]{n \vdash \gamma} b \dashv \left(\begin{array}{c} \langle \langle P' \rangle \rangle \\ \langle \langle Q' \rangle \rangle \end{array} \right).$$

- For the rule

$$\frac{n + 1 \vdash P}{n \vdash \nu a_{n+1}(P)}$$

we observe that by the rightmost rule in (??), $\nu a_{n+1}(P) \xrightarrow{\alpha} P'$ iff $P' = \nu a_{n+1}(P'')$ and $P \xrightarrow{\beta} P''$ with $\beta \setminus \{a_{n+1}\} = \alpha$. By the induction hypothesis $n + 1 \dashv \langle \langle P \rangle \rangle \xrightarrow[n \bullet]{n + 1 \vdash \beta} b \dashv \langle \langle P'' \rangle \rangle$. Now $\langle \langle n \vdash \nu a_{n+1}(P) \rangle \rangle$ is equal by definition to $n \dashv \langle \langle P \rangle \rangle$. Since $n \dashv \left(\begin{array}{c} n \\ n + 1 \vdash \beta \end{array} \xrightarrow[n \bullet]{n + 1 \vdash \beta} b \dashv \langle \langle P'' \rangle \rangle \right)$, then

$$n \dashv \langle \langle P \rangle \rangle \xrightarrow[n \bullet]{n + 1 \vdash \beta} b \dashv \langle \langle P'' \rangle \rangle = n + 1 \dashv \langle \langle P'' \rangle \rangle.$$

- For the rule

$$\frac{ar(\mathbf{f}) = n}{n \vdash \mathbf{f}}$$

519
520

the result is immediate by the definition of β_b and the two leftmost rules in (??).

- For the rule

$$\frac{n \vdash P \quad \text{degree}(\sigma) \leq n}{n \vdash P\sigma}$$

we observe that by the rightmost rule in (??), $P\sigma \xrightarrow{\alpha} P'$ iff $P' = P''\sigma$ and $P' \xrightarrow{\alpha \circ \sigma^{-1}} P''$. By the induction hypothesis, we have that $n \dashv \langle \langle P \rangle \rangle \xrightarrow[n \bullet]{n \vdash \alpha \circ \sigma^{-1}} b \dashv \langle \langle P'' \rangle \rangle$. Observe that $n \vdash \alpha \circ \sigma^{-1}$, since $n \vdash \alpha$ and both σ and σ^{-1} have degree n . Now, $\langle \langle n \vdash P\sigma \rangle \rangle$ is equal by definition to $n \dashv \left(\begin{array}{c} \sigma \\ \langle \langle P \rangle \rangle \end{array} \right)$. Since $n \dashv \left(\begin{array}{c} \sigma \\ n \end{array} \xrightarrow[n \bullet]{n \vdash \alpha \circ \sigma^{-1}} b \dashv \langle \langle P'' \rangle \rangle \right)$, then

$$n \dashv \left(\begin{array}{c} \sigma \\ \langle \langle P \rangle \rangle \end{array} \right) \xrightarrow[n \bullet]{n \vdash \alpha \circ \sigma^{-1}} b \dashv \langle \langle P'' \rangle \rangle = n \dashv \langle \langle P'' \rangle \rangle.$$

- For the rule

$$\frac{n \vdash P}{n + 1 \vdash P}$$

observe that for all processes P , if $P \xrightarrow{\alpha}_H P'$, then $al(\alpha) \subseteq al(P)$. Since $n \vdash P$, $al(P) \subseteq \{a_1, \dots, a_n\}$, then $al(\alpha) \subseteq \{a_1, \dots, a_n\}$. So $n \vdash \alpha$. By the induction hypothesis $\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle P' \rangle\rangle$. Now, $\langle\langle n+1 \vdash P \rangle\rangle$ is equal by definition to $\frac{n}{\bullet} \langle\langle P \rangle\rangle$. Since $\frac{n}{\bullet} \xrightarrow{\frac{n+1+\alpha}{n+\alpha}}_b \frac{n}{\bullet}$, then

$$\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+1+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle P' \rangle\rangle .$$

521

□

522 **Proposition 19.** *Let $n \vdash P$, $n \vdash \alpha$ such that $al(\alpha) \subseteq al(P)$*

523 1. *If $\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d \rangle\rangle$, then there exists $n \vdash P'$ such that $P \xrightarrow{\alpha}_H P'$*
 524 *and $\frac{n}{\bullet} \langle\langle P' \rangle\rangle = \frac{n}{\bullet} \langle\langle d \rangle\rangle$.*

525 2. *If $\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_w \frac{n}{\bullet} \langle\langle d \rangle\rangle$ then there is $n \vdash P'$ such that $P \xrightarrow{\alpha}_M P'$ and*
 526 *$\frac{n}{\bullet} \langle\langle P' \rangle\rangle = \frac{n}{\bullet} \langle\langle d \rangle\rangle$.*

527 *Proof.* We prove 1, the proof for 2 is analogous. As before, given $n \vdash \alpha$, we
 528 write $n+n \vdash \alpha \oplus \alpha$ for the function mapping a_i to $\alpha(a_i)$ if $i \leq n$ and to $\alpha(a_{i-n})$
 529 if $i > n$.

530 Suppose that $\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d \rangle\rangle$. We prove by induction on the rules
 531 of (??) that there exists $n \vdash P'$ such that $P \xrightarrow{\alpha}_H P'$ and $\frac{n}{\bullet} \langle\langle P' \rangle\rangle = \frac{n}{\bullet} \langle\langle d \rangle\rangle$.

- For the rule

$$\frac{n \vdash P \quad n \vdash Q}{n \vdash P|Q}$$

we observe that by definition $\langle\langle n \vdash P|Q \rangle\rangle = \frac{n}{\bullet} \langle\langle P \rangle\rangle \langle\langle Q \rangle\rangle$. Since

$$\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d \rangle\rangle \quad \text{and} \quad \frac{n}{\bullet} \langle\langle Q \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d \rangle\rangle$$

then $\frac{n}{\bullet} \langle\langle P \rangle\rangle \langle\langle Q \rangle\rangle \xrightarrow{\alpha}_b \frac{n}{\bullet} \langle\langle d \rangle\rangle$ iff

$$\frac{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d_1 \rangle\rangle \quad \text{and} \quad \frac{n}{\bullet} \langle\langle Q \rangle\rangle \xrightarrow{\frac{n+\alpha}{\bullet}}_b \frac{n}{\bullet} \langle\langle d_2 \rangle\rangle$$

532 with $\overset{n}{\vdash} d = \overset{n}{\bullet} \left(\begin{array}{c} \boxed{d_1} \\ \boxed{d_2} \end{array} \right)$. Now take $\beta' = \alpha \cap al(Q)$ and $\alpha' = \alpha \cap al(P)$.

533 We have that $\beta' \subseteq al(Q)$ and $\alpha' \subseteq al(P)$. Moreover, since $\alpha \subseteq al(P) \cup$
 534 $al(Q)$ by hypothesis, it holds that $\alpha' \cup \beta' = \alpha$ and that $\alpha' \cap al(Q) =$
 535 $\beta' \cap al(P)$.

By Lemma ?? and $\overset{n}{\vdash} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{\bullet} \overset{n}{\vdash} d_1$ and $\overset{n}{\vdash} \langle\langle Q \rangle\rangle \xrightarrow[n+\alpha]{\bullet} \overset{n}{\vdash} d_2$,

we obtain that $\overset{n}{\vdash} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha']{\bullet} \overset{n}{\vdash} d_1$ and $\overset{n}{\vdash} \langle\langle Q \rangle\rangle \xrightarrow[n+\beta']{\bullet} \overset{n}{\vdash} d_2$

with $\overset{n}{\vdash} d = \overset{n}{\bullet} \left(\begin{array}{c} \boxed{d_1} \\ \boxed{d_2} \end{array} \right)$. We can now use induction hypothesis to get

a P' and Q' such that $\overset{n}{\vdash} \langle\langle P' \rangle\rangle = \overset{n}{\vdash} d_1$, $\overset{n}{\vdash} \langle\langle Q' \rangle\rangle = \overset{n}{\vdash} d_2$,

$P \xrightarrow{\alpha'} P'$ and $Q \xrightarrow{\beta'} Q'$. By the leftmost rule in (??), we have that

$$P|Q \xrightarrow{\alpha} P'|Q'.$$

Finally, by definition of $\langle\langle \cdot \rangle\rangle$,

$$\langle\langle n \vdash P'|Q' \rangle\rangle = \overset{n}{\bullet} \left(\begin{array}{c} \langle\langle P' \rangle\rangle \\ \langle\langle Q' \rangle\rangle \end{array} \right) = \overset{n}{\bullet} \left(\begin{array}{c} \boxed{d_1} \\ \boxed{d_2} \end{array} \right) = \overset{n}{\vdash} d.$$

- For the rule

$$\frac{n+1 \vdash P}{n \vdash \nu a_{n+1}(P)}$$

we observe that by definition $\langle\langle n \vdash \nu a_{n+1}P \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle$. Since

$\overset{n}{\bullet} \xrightarrow[n+1+\beta]{n+\alpha} \overset{n}{\bullet}$ with $\alpha = \beta \setminus \{a_{n+1}\}$, then $\overset{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow{\alpha} \overset{n}{\vdash} d$

iff $\overset{n}{\vdash} \langle\langle P \rangle\rangle \xrightarrow[n+1+\beta]{\bullet} \overset{n}{\vdash} d'$ with $\overset{n}{\vdash} d = \overset{n}{\bullet} d'$. We can now

use the induction hypothesis to get a P' such that $\overset{n}{\vdash} \langle\langle P' \rangle\rangle = \overset{n}{\vdash} d'$

and $P \xrightarrow{\alpha} P'$. By the fact that $\alpha = \beta \setminus \{a_{n+1}\}$ and the rightmost rule in (??), we have that

$$\nu a_{n+1}(P) \xrightarrow{\alpha} \nu a_{n+1}(P').$$

By definition of $\langle\langle \cdot \rangle\rangle$,

$$\langle\langle n \vdash \nu a_{n+1}(P') \rangle\rangle = \overset{n}{\bullet} \langle\langle P' \rangle\rangle = \overset{n}{\bullet} d' = \overset{n}{\vdash} d.$$

- For the rule

$$\frac{ar(\mathbf{f}) = n}{n \vdash \mathbf{f}}$$

the result is immediate by the definition of β_b and the two leftmost rules in (??).

- For the rule

$$\frac{n \vdash P \quad \text{degree}(\sigma) \leq n}{n \vdash P\sigma}$$

we observe that by definition $\langle\langle n \vdash P\sigma \rangle\rangle = \overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P \rangle\rangle$. Since $\bar{\sigma} \xrightarrow[n+\alpha \circ \sigma^{-1}]{} \bar{\sigma}$, then $\overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{} \overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P \rangle\rangle$ iff $\overset{n}{\square} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha \circ \sigma^{-1}]{} \overset{n}{\square} \langle\langle P \rangle\rangle$ with $\overset{n}{\square} \langle\langle P \rangle\rangle = \overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P \rangle\rangle$. We can now use the induction hypothesis to get a P' such that $\overset{n}{\square} \langle\langle P' \rangle\rangle = \overset{n}{\square} \langle\langle P \rangle\rangle$ and $P \xrightarrow{\alpha \circ \sigma^{-1}} P'$. By the rightmost rule in (??), we have that

$$P\sigma \xrightarrow{\alpha} P'\sigma$$

and, by the definition of $\langle\langle \cdot \rangle\rangle$,

$$\langle\langle n \vdash P'\sigma \rangle\rangle = \overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P' \rangle\rangle = \overset{n}{\square} \overset{\bar{\sigma}}{\square} \overset{n}{\square} \langle\langle P \rangle\rangle = \overset{n}{\square} \langle\langle P \rangle\rangle.$$

- For the rule

$$\frac{n \vdash P}{n+1 \vdash P}$$

we observe that since $n \vdash P$, then $al(P) \subseteq \{a_1, \dots, a_n\}$ and thus $a_{n+1} \notin \alpha$. By definition, $\langle\langle n+1 \vdash P \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle$. Since $\overset{n}{\bullet} \xrightarrow[n+\alpha]{} \overset{n}{\bullet}$, then $\overset{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow[n+1+\alpha]{} \overset{n}{\bullet} \langle\langle P \rangle\rangle$ iff $\overset{n}{\bullet} \langle\langle P \rangle\rangle \xrightarrow[n+\alpha]{} \overset{n}{\bullet} \langle\langle P \rangle\rangle$ with $\overset{n}{\bullet} \langle\langle P \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle$. We can now use induction hypothesis to get a P' such that $\overset{n}{\bullet} \langle\langle P' \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle$ and

$$P \xrightarrow{\alpha} P'.$$

By definition of $\langle\langle \cdot \rangle\rangle$,

$$\langle\langle n+1 \vdash P' \rangle\rangle = \overset{n}{\bullet} \langle\langle P' \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle = \overset{n}{\bullet} \langle\langle P \rangle\rangle.$$

□

Lemma 20. Let $n \vdash P$, $n \vdash \alpha$, $i \leq n$ and $a_i \notin al(P)$.

$$\langle\langle n \vdash P \rangle\rangle \xrightarrow[n+\alpha]{} \langle\langle n \vdash Q \rangle\rangle \quad \text{iff} \quad \langle\langle n \vdash P \rangle\rangle \xrightarrow[n+\alpha \cup \{a_i\}]{} \langle\langle n \vdash Q \rangle\rangle.$$

Proof. First, using Lemma ??, we can find $n \multimap^1 d$ such that

$$n \multimap \langle\langle P \rangle\rangle = \frac{i-1}{n-i} \multimap d$$

Then, we use a permutation to relocate the \multimap : let σ be the transposition $(i \ n)$. By definition of $\langle\langle - \rangle\rangle$ we have

$$n \multimap \langle\langle P\sigma \rangle\rangle = n \multimap \bar{\sigma} \multimap \langle\langle P \rangle\rangle = \frac{n-1}{\bullet} \multimap d$$

From the derivation rules for props (§ ??), we see that any transition out of $n \multimap \langle\langle P\sigma \rangle\rangle$ must come from the derivation rule for \oplus , namely:

$$\frac{n \multimap^1 d \xrightarrow{\alpha} b \quad n \multimap^1 e \quad \multimap \xrightarrow{v} b \quad \bullet}{n \multimap^1 d \xrightarrow{\alpha v} b \quad n \multimap^1 e} \lambda^3$$

Hence, we must have $n \multimap \langle\langle Q\sigma \rangle\rangle = \frac{n-1}{\bullet} \multimap e$ for some $n \multimap e$. Now, notice that we have $\multimap \xrightarrow{1} b \quad \bullet$ and $\multimap \xrightarrow{0} b \quad \bullet$ so that

$$n \multimap \langle\langle P\sigma \rangle\rangle \xrightarrow{n \vdash \alpha} b \quad n \multimap \langle\langle Q\sigma \rangle\rangle \quad \text{iff} \quad n \multimap \langle\langle P\sigma \rangle\rangle \xrightarrow{n \vdash \alpha \cup \{a_n\}} b \quad n \multimap \langle\langle Q\sigma \rangle\rangle.$$

Applying the transposition $(i \ n)$ again we can conclude that

$$n \multimap \langle\langle P \rangle\rangle \xrightarrow{n \vdash \alpha} b \quad n \multimap \langle\langle Q \rangle\rangle \quad \text{iff} \quad n \multimap \langle\langle P \rangle\rangle \xrightarrow{n \vdash \alpha \cup \{a_i\}} b \quad n \multimap \langle\langle Q \rangle\rangle.$$

539

□

Lemma 21. *Let $n \vdash P$, $i \leq n$ and $a_i \notin al(P)$. Then there exists $n \multimap^1 d$ such that*

$$n \multimap \langle\langle P \rangle\rangle = \frac{i-1}{n-i} \multimap d$$

540 *Proof.* We prove this by structural induction on the rules of (??).

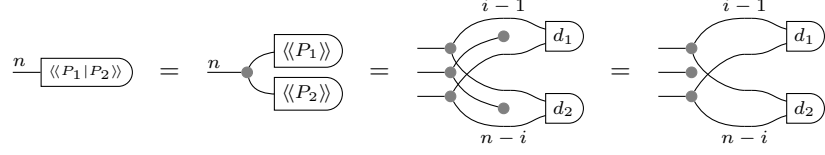
- For the rule

$$\frac{n \vdash P_1 \quad n \vdash P_2}{n \vdash P_1 | P_2}$$

$al(P_1 | P_2) = al(P_1) \cup al(P_2)$ and therefore $a_i \notin al(P)$ and $a_i \notin al(P_2)$. We can apply the induction hypothesis to obtain d_1 and d_2 such that

$$n \multimap \langle\langle P_1 \rangle\rangle = \frac{i-1}{n-i} \multimap d_1 \quad \text{and} \quad n \multimap \langle\langle P_2 \rangle\rangle = \frac{i-1}{n-i} \multimap d_2$$

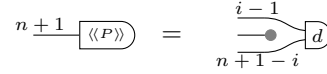
and we can deduce



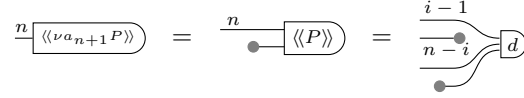
- For the rule

$$\frac{n+1 \vdash P}{n \vdash \nu a_{n+1}(P)}$$

we have $al(\nu a_{n+1}(P)) = al(P) \setminus \{a_{n+1}\}$ which implies that $a_i \notin al(P)$. We can therefore apply the induction hypothesis to find d such that



and we can deduce

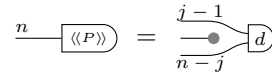


541
542

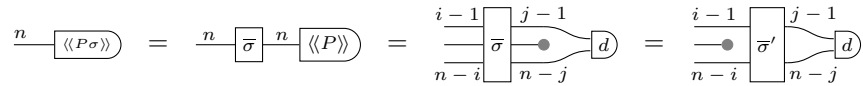
- P cannot be equal to a process variable \mathbf{f} since, for $n \vdash \mathbf{f}$ we have $al(\mathbf{f}) = ar(\mathbf{f}) = n$.
- For the rule

$$\frac{n \vdash P \quad degree(\sigma) \leq n}{n \vdash P\sigma}$$

we have $al(P\sigma) = \sigma[al(P)]$. Therefore $a_j \notin al(P)$ for $j := \sigma^{-1}(i)$. By the induction hypothesis, we can find d such that



and conclude that



543

for some permutation $\sigma' : n-1 \rightarrow n-1$.

- For the rule

$$\frac{n + 1 \vdash P}{n \vdash \nu a_{n+1}(P)}$$

we have $n \dashv \langle\langle P \rangle\rangle = \overset{n-1}{\dashv \bullet} \langle\langle P \rangle\rangle$ and, using the induction hypothesis we can find d such that

$$n \dashv \langle\langle P \rangle\rangle = \overset{i-1}{\dashv \bullet} \langle\langle P \rangle\rangle \dashv d$$

544 The conclusion follows immediately.

545

□

546 7. Related and Future Work

547 The terminology *Hoare and Milner synchronisation* is used in Synchronised
 548 Hyperedge Replacement (SHR) [? ?]. Our work is closely related to SHR:
 549 indeed, the prop $\mathcal{S}_{\text{CW}}(\Sigma)$ has arrows open hypergraphs, where hyperedges are
 550 labeled with elements of Σ [?]. To define a coalgebra $\beta: \Sigma \rightarrow \mathcal{F}\mathcal{S}_{\text{CW}}(\Sigma)$ is to
 551 specify a transition system for each label in Σ . Then, constructing the coalgebra
 552 $\beta^\sharp: \mathcal{S}_{\text{CW}}(\Sigma) \rightarrow \mathcal{F}\mathcal{S}_{\text{CW}}(\Sigma)$ from a distributive law amounts to giving a transition
 553 system to all hypergraphs according to some synchronisation policy (e.g. à la
 554 Hoare or à la Milner). SHR systems equipped with Hoare and Milner synchronisation
 555 are therefore instances of our approach. A major difference is our focus
 556 on the algebraic aspects: e.g. since string diagrams can be regarded as syntax as
 557 well as combinatorial entities, their syntactic nature allows for the bialgebraic
 558 approach, and simple inductive proofs. The operational rules in Figure ?? are
 559 also those of tile systems [?]. However, in the context of tiles, transitions are
 560 arrows of the vertical *category*: this forces every state to perform at least one
 561 identity transition. For example, it is not possible to consider empty sets of
 562 transitions, which can be a useful feature in the string diagrammatic approach,
 563 see [?].

564 Amongst the many other related models, it is worth mentioning bigraphs [?
 565]. While also graphical, bigraphs can be nested hierarchically, a capability that
 566 we have not considered. Moreover, the behaviour functor \mathcal{F} in § ?? forces
 567 the labels and the arriving states to have the same sort as the starting states.
 568 Therefore, fundamental mobility mechanisms such as scope-extrusion cannot
 569 immediately be addressed within our framework. We are confident, however,
 570 that the solid algebraic foundation we have laid here for the operational semantics
 571 of two-dimensional syntax will be needed to shed light on such concepts as
 572 hierarchical composition and mobility. Some ideas may come from [?].

573 **Appendix A. An Example of Bialgebraic Semantics: Non-Deterministic**
574 **Automata**

575 In this appendix, we illustrate bialgebraic semantics on a well-known case
576 study, namely non-deterministic automata [?]. The intention is to provide in
577 full a basic example that may serve as a roadmap for the approach to string
578 diagrams proposed in the main text.

579 **Deterministic automata as coalgebras.** Let 2 be the set $\{0, 1\}$ and A an
580 alphabet of symbols. A deterministic automata (DA) is a pair $(S, \langle o, t \rangle)$ where S
581 is a set of state and $\langle o, t \rangle: S \rightarrow 2 \times S^A$ consists of the *output function* $o: S \rightarrow 2$,
582 defining whether a state $x \in S$ is accepting ($o(x) = 1$) or not ($o(x) = 0$), and
583 the *transition function* $t: S \rightarrow S^A$ mapping each state $x \in S$ and each $a \in A$
584 the successor state $t(x)(a)$.

585 DAs are in one to one correspondence with coalgebras for the functor $\mathcal{F}: \mathbf{Set} \rightarrow$
586 \mathbf{Set} defined as $\mathcal{F}(X) = 2 \times X^A$. The set of all languages over the alphabet A ,
587 hereafter denoted by 2^{A^*} , carries a final coalgebra. For each deterministic au-
588 tomaton $(S, \langle o, t \rangle)$, there is a unique coalgebra homomorphism $\llbracket \cdot \rrbracket: S \rightarrow 2^{A^*}$
589 assigning to each state in x the language that it accepts (defined for all words
590 $w \in A^*$ as $\llbracket x \rrbracket(\epsilon) = o(x)$ and $\llbracket x \rrbracket(aw) = \llbracket t(x)(a) \rrbracket(w)$).

591 **Non deterministic automata give rise to bialgebras.** A non deterministic
592 automata (NDA) is a pair $(S, \langle o, t \rangle)$ where S and o are like for deterministic
593 automata, but the transition function t has now type $S \rightarrow \mathcal{P}_\omega S^A$, where \mathcal{P}_ω is
594 the finite powerset functor. t maps each state $x \in S$ and each $a \in A$ to a (finite)
595 set of possible successor states $t(x)(a)$.

Therefore NDA are coalgebras for the functor $\mathcal{F}\mathcal{P}_\omega$ where \mathcal{F} is the functor for
deterministic automata explained above. Traditionally the semantics of NDA
is also defined in terms of languages, but the set 2^{A^*} does not carry a final
coalgebra for $\mathcal{F}\mathcal{P}_\omega$. The solution is to view NDAs as DAs, by a construction that
in automata theory is usually called determinisation (or powerset construction).
Categorically, this amounts to transform an NDA $\beta: S \rightarrow 2 \times \mathcal{P}_\omega(S)^A$ into a
DA $\beta^\sharp: \mathcal{P}_\omega(S) \rightarrow 2 \times \mathcal{P}_\omega(S)^A$, i.e. an $\mathcal{F}\mathcal{P}_\omega$ -coalgebra into an \mathcal{F} -coalgebra. For
the determinised NDA β^\sharp , there exists a unique \mathcal{F} -coalgebra homomorphism
 $\llbracket \cdot \rrbracket: \mathcal{P}_\omega(S) \rightarrow 2^{A^*}$. This yields language semantics $S \rightarrow 2^{A^*}$ for the original
statespace S , by precomposing with the unit for the monad $\eta_S: S \rightarrow \mathcal{P}_\omega(S)$, as
in the diagram below.

$$\begin{array}{ccc}
S & \xrightarrow{\eta} & \mathcal{P}_\omega(S) \dashrightarrow \llbracket \cdot \rrbracket \dashrightarrow 2^{A^*} \\
\downarrow \beta & \searrow \beta^\sharp & \downarrow \\
\mathcal{F}\mathcal{P}_\omega(S) & \xrightarrow{\mathcal{F}(\llbracket \cdot \rrbracket)} & \mathcal{F}\mathcal{P}_\omega(2^{A^*})
\end{array} \tag{A.1}$$

In automata theory, there is a concrete way of constructing β^\sharp from β . In the
categorical abstraction, the general principle underpinning this construction is

that defining β^\sharp requires the introduction of a *distributive law* $\lambda: \mathcal{P}_\omega \mathcal{F} \Rightarrow \mathcal{F} \mathcal{P}_\omega$. For all sets X , $\lambda_X: \mathcal{P}_\omega(2 \times S^A) \rightarrow 2 \times \mathcal{P}_\omega(X)^A$ is defined for any $b_1, \dots, b_n \in 2$ and for all $\phi_1, \dots, \phi_n \in S^A$,

$$\lambda_X(\{\langle b_1, \phi_1 \rangle, \dots, \langle b_n, \phi_n \rangle\}) = \langle \bigsqcup_{i \in 1 \dots n} b_i, \bigsqcup_{i \in 1 \dots n} \phi_i \rangle \quad (\text{A.2})$$

596 where the leftmost \sqcup is the obvious join in 2 (regarded as the partial order
597 $0 \sqsubseteq 1$), and the rightmost one is defined as the point-wise union: namely for all
598 $a \in A$, $\bigsqcup_{i \in 1 \dots n} \phi_i(a) = \{\phi_1(a), \dots, \phi_n(a)\}$. Observe that, in the particular case
599 of the empty set \emptyset , $\alpha(\emptyset) = \langle 0, \phi_\emptyset \rangle$ where $\phi_\emptyset(a) = \emptyset$ for all $a \in A$.

600 Given λ , β^\sharp is defined as $\mathcal{P}_\omega(S) \xrightarrow{\mathcal{P}_\omega \beta} \mathcal{P}_\omega \mathcal{F} \mathcal{P}_\omega(S) \xrightarrow{\lambda_{\mathcal{P}_\omega(S)}} \mathcal{F} \mathcal{P}_\omega \mathcal{P}_\omega(S) \xrightarrow{\mathcal{F} \mu} \mathcal{F} \mathcal{P}_\omega(S)$. The reader could also check that, with this definition, β^\sharp is a λ -
601 bialgebra, where the algebraic structure on the state space $\mathcal{P}_\omega(S)$ is given by
602 the multiplication $\mu_S: \mathcal{P}_\omega \mathcal{P}_\omega(S) \rightarrow \mathcal{P}_\omega(S)$.
603

Algebraic presentation of \mathcal{P}_ω . In concrete, determinisation is about obtain-
ing the language semantics of NDAs. The essence of its categorical abstraction
is moving from coalgebras to bialgebras, thus taking into account the *algebraic*
structure of the statespace $\mathcal{P}_\omega(S)$. Indeed, it is well-known that the monad
 \mathcal{P}_ω is presented by the algebraic theory of join semilattice with bottom: these
consist of a set S equipped with an operation $\otimes: S \times S \rightarrow S$ and an element
 $\mathbf{0}: 1 \rightarrow S$ such that

$$(x \otimes y) \otimes z = x \otimes (y \otimes z) \quad x \otimes y = y \otimes x \quad x \otimes x = x \quad x \otimes \mathbf{0} = x \quad (\text{A.3})$$

604 for all $x, y, z \in S$. To make formal the link with \mathcal{P}_ω , let us introduce the
605 functor \mathcal{T}_S which maps every set S to the set $\mathcal{T}_S(S)$ of terms built with \otimes ,
606 $\mathbf{0}$ and variables in S . This functor carries the structure of a monad, where
607 the unit $\eta_S: S \rightarrow \mathcal{T}_S(S)$ is given by inclusion (each variable is a term) and
608 the multiplication $\mu_S: \mathcal{T}_S \mathcal{T}_S(S) \rightarrow \mathcal{T}_S(S)$ by substitution. The fact that \mathcal{P}_ω is
609 presented by the algebraic theory of join semilattice with bottom means that
610 \mathcal{P}_ω is (isomorphic to) the quotient of the monad \mathcal{T}_S by equations (??).

611 **Modular construction of the distributive law.** The monad \mathcal{P}_ω is associ-
612 ated with a rather elementary algebraic theory, which allows λ to be defined in a
613 rather straightforward way. The monads that we use in the main text to encode
614 categorical structures are way more involved. For this reason, it is important
615 to modularise the task of finding a distributive law. Again, we are going to use
616 $\lambda: \mathcal{P}_\omega \mathcal{F} \Rightarrow \mathcal{F} \mathcal{P}_\omega$ in (??) as a proof of concept, and show how its definition can
617 be divided into different stages.

618 By the above discussion, \mathcal{P}_ω is the quotient of \mathcal{T}_S by equations (??), but we
619 can decompose this even further: \mathcal{T}_S is the free monad (see § ??) $(\mathcal{S}_1 + \mathcal{S}_2)^\dagger$ over
620 the endofunctor expressing the semilattice signature: here $\mathcal{S}_1: S \mapsto \{\mathbf{0}\}$ stands
621 for the element $\mathbf{0}$, and $\mathcal{S}_2: S \mapsto S \times S$ encodes the operation \otimes .

622 We can now divide the construction of λ in three stages:

Distributive law on the signature. The first step is to give separate dis-
tributive laws of \mathcal{S}_1 and \mathcal{S}_2 over \mathcal{F} . The first one is $\lambda^1: \mathcal{S}_1 \mathcal{F} \Rightarrow \mathcal{F} \mathcal{S}_1$.

Given a set S , we let $\lambda_S^1: \{\mathbf{0}\} \rightarrow 2 \times \{\mathbf{0}\}^A$ map $\mathbf{0}$ to the pair $\langle 0, ! \rangle$ where $!$ is the unique function of type $A \rightarrow \{\mathbf{0}\}$. This distributive law can also be conveniently presented as the following pair of derivation rules. The first describes the element 0 ($= \mathbf{0}$ is not a terminating state) of the pair $\lambda_S^1\{\mathbf{0}\}$; the second describes the element $! : A \rightarrow \{\mathbf{0}\}$ of the pair $\lambda_S^1\{\mathbf{0}\}$.

$$\frac{}{\mathbf{0} \Downarrow} \qquad \frac{a \in A}{\mathbf{0} \xrightarrow{a} \mathbf{0}} \qquad (\text{A.4})$$

The second distributive law is $\lambda^1: \mathcal{S}_2\mathcal{F} \rightarrow \mathcal{F}\mathcal{S}_2$. Given a set S , we let $\lambda_S^2: (2 \times S^A) \times (2 \times S^A) \rightarrow 2 \times (S \times S)^A$ be defined for all $\langle b_1, \phi_1 \rangle, \langle b_2, \phi_2 \rangle \in 2 \times S^A$ as $\lambda^2(\langle b_1, \phi_1 \rangle, \langle b_2, \phi_2 \rangle) = \langle b_1 \sqcup b_2, \langle \phi_1, \phi_2 \rangle \rangle$. The representation in terms of derivation rules is:

$$\frac{s_1 \downarrow}{s_1 \otimes s_2 \downarrow} \qquad \frac{s_2 \downarrow}{s_1 \otimes s_2 \downarrow} \qquad \frac{s_1 \xrightarrow{a} s'_1 \quad s_2 \xrightarrow{a} s'_2}{s_1 \otimes s_2 \xrightarrow{a} s'_1 \otimes s'_2} \qquad (\text{A.5})$$

623 **Distributive law on the free monad.** We can now put together λ^1 and λ^2
624 to obtain a distributive law of type $\mathcal{T}_S\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}_S$. First, by post-
625 composing λ^1 and λ^2 with the unit of the free monads $\eta: \mathcal{S}_i \Rightarrow \mathcal{S}_i^\dagger$
626 associated with endofunctors \mathcal{S}_i , one obtains two GSOS specifications
627 $\bar{\lambda}^1: \mathcal{S}_1\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_1^\dagger$ and $\bar{\lambda}^2: \mathcal{S}_2\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}_2^\dagger$. As explained in Section ??, one
628 can take the coproduct of GSOS specification, so to obtain $\bar{\lambda}: \mathcal{S}\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}^\dagger$
629 and thus, by Proposition ??, a distributive law $\bar{\lambda}^\dagger: \mathcal{S}^\dagger\mathcal{F} \Rightarrow \mathcal{F}\mathcal{S}^\dagger$. Since
630 $\mathcal{S}^\dagger = \mathcal{T}_S$, this distributive law is effectively of the desired type $\mathcal{T}_S\mathcal{F} \Rightarrow$
631 $\mathcal{F}\mathcal{T}_S$.

632 **Distributive law on the quotient.** As \mathcal{P}_ω is a quotient of \mathcal{T}_S by equations
633 (??), the last step is quotienting the $\bar{\lambda}^\dagger: \mathcal{T}_S\mathcal{F} \Rightarrow \mathcal{F}\mathcal{T}_S$ by such equations.
634 There is a categorical approach, described in § ??, which allows to turn
635 $\bar{\lambda}^\dagger$ into a distributive law λ on the quotient of \mathcal{T}_S , provided that $\bar{\lambda}^\dagger$ is
636 compatible with equations (??). This is indeed the case for $\bar{\lambda}^\dagger$, which thus
637 yields by Proposition ?? a distributive law $\lambda: \mathcal{P}_\omega\mathcal{F} \Rightarrow \mathcal{F}\mathcal{P}_\omega$ of the desired
638 type. The definition (??) of λ given above can be equivalently described
639 by the derivation rules (??)-(??), modulo the algebraic representation of
640 \mathcal{P}_ω in terms of join semilattices.