Relational Semantics of Non-Deterministic Dataflow

Thomas Troels Hildebrandt
Prakash Panangaden
Glynn Winskel

BRICS Report Series RS-97-36

ISSN 0909-0878 December 1997

Relational Semantics of Non-Deterministic Dataflow

(Extended Abstract)

Thomas Hildebrandt[†] Prakash Panangaden[‡] Glynn Winskel[†]

[†]BRICS,[*] University of Aarhus, Denmark
[‡]McGill University, Canada

Abstract

We recast dataflow in a modern categorical light using profunctors as a generalization of relations. The well known causal anomalies associated with relational semantics of indeterminate dataflow are avoided, but still we preserve much of the intuitions of a relational model. The development fits with the view of categories of models for concurrency and the general treatment of bisimulation they provide. In particular it fits with the recent categorical formulation of feedback using traced monoidal categories. The payoffs are: (1) explicit relations to existing models and semantics, especially the usual axioms of monotone IO automata are read off from the definition of profunctors, (2) a new definition of bisimulation for dataflow, the proof of the congruence of which benefits from the preservation properties associated with open maps and (3) a treatment of higher-order dataflow as a biproduct, essentially by following the geometry of interaction programme.

1 Introduction

Our background includes work done on presenting models for concurrency as categories, as summarised in [42]. This enabled a sweeping definition of bisimulation based on open maps applicable to any category of models equipped with a distinguished subcategory of paths [19]. It also exposed a new space of models. Presheaf categories possess a canonical choice of open maps and bisimulation, and can themselves be related in the bicategory of profunctors. This yields a form of domain theory but boosted to the level of using categories rather than partial orders as the appropriate domains.

One argument for the definition of bisimulation based on open maps is the powerful preservation properties associated with it. Notable is the result of [8] that any colimit preserving functor between presheaf categories preserves bisimulation,

[*]**B**asic **R**esearch **in** **C**omputer **S**cience, Centre of the Danish National Research Foundation.

which besides obvious uses in relating semantics in different models with different notions of bisimulation is, along with several other general results, useful in establishing congruence properties of process languages. By understanding dataflow in terms of profunctors we are able to exploit the framework not just to give a definition of bisimulation between dataflow networks but also in showing it to be a congruence with respect to the standard operations of dataflow.

A general definition of bisimulation is all well and good but it needs to be tested and its consequences understood for a range of process languages. Another argument in favour of the presheaf approach to bisimulation is that when tried against traditional process languages it yields persuasive results, as in [8, 40, 7]. But still these are just examples and it is hoped that a more satisfying and conclusive argument will come from an endeavour to ascertain the operational content of presheaf models more generally.

One difficulty has been in understanding the operational significance of the bisimulation which comes from open maps for higher-order process languages (where for example processes themselves can be passed as values). Another gap, more open and so more difficult to approach, is that whereas both interleaving models and independence models like event structures can be recast as presheaf models, as soon as higher-order features appear, the presheaf semantics at present reduce concurrency to nondeterministic interleaving. A study of nondeterministic dataflow is helpful here as its compositional models are forced to account for causal dependency using ideas familiar from independence models; at the same time the models are a step towards understanding higher-order as they represent nondeterministic functions from input to output.

It is notable that the profunctor semantics of dataflow yields automatically the axioms for monotone port automata used in modelling dataflow [30] in contrast to the work in [38]. At the same time we have to work to get a correct operation on profunctors to model the dataflow feedback; "the obvious" choice of modelling feedback by coend doesn't account for the subtle causal constraints which plague dataflow semantics.

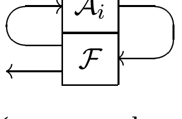
The idea that non-deterministic dataflow can be modelled by some kind of generalised relations fits with that of others, notably Stark in [38, 39]. That dataflow should fit within a categorical account of feedback accords for instance with [23, 1]. But in presenting a semantics of dataflow as profunctors we obtain the benefits to be had from placing nondeterministic dataflow centrally within categories of models for concurrency, and in particular within presheaf models. One of our future aims is a dataflow semantics of the hardware-description language Verilog HDL [14], which presently only possesses a noncompositional, operational definition. The semantics of a language of this richness requires a flexible yet abstract domain theory of the kind presheaf models seem able to support.

2 Models for indeterminate dataflow

The Dataflow paradigm for asynchronous parallel computation, originated in work of Jack Dennis and others in the mid-sixties [21, 10, 11]. The basic idea is that data flows between autonomous computing agents, that are interconnected by communication channels. The essential idea is that computation is triggered by the arrival of data rather than by flow of control. The channels are assumed to act as unbounded FIFO-queues. For dataflow networks built from only *deterministic* nodes, Kahn [21] has argued that their behaviour could be captured *denotationally* in a very simple and elegant fashion, using elementary domain theory, which is later shown formally by several authors, e.g. Faustini [13], Lynch and Stark [26]. The key idea is to model the behaviour of each port a as a stream of values. A node can then be modelled as a continuous function between such streams and the combined network as a least fixed point of a set of equations describing the components. In this sense Kahn's semantics is compositional. Subsequently, different semantics have been described as satisfying *Kahn's principle* when they are built up compositionally along similar lines. Note that the *observable behaviour* is taken to be the *input-output* relation between completed sequences of values, thus it completely abstracts away from causal dependencies between values on different ports.

2.1 The need for causality

For *indeterminate* networks, the situation is not so simple. Brock and Ackerman[6] showed that for networks containing the nondeterministic primitive *fair merge*, the input-output relations are not compositional, ie. if we simply choose the input-output relation as observable behaviour, we cannot define a compositional semantics, which is adequate with respect to the operational semantics. Hence a straightforward generalisation of Kahn's model to get a compositional semantics for these indeterminate networks fails. Later, Traktenbrot and Rabinovich, and independently, Russell showed, that even for the simplest nondeterministic primitive the ordinary *bounded choice* (or "unfair merge"), the input-output relation is not compositional. We present an example close to that of Traktenbrot and Rabinovich. It works by giving two simple examples of automata $\mathcal{A}_1$ and $\mathcal{A}_2$, which have the same input-output relation, and a context in which they behave differently, pictorially



.

The context is a fork process $\mathcal{F}$ (a process that copies every input to two outputs), through which the output of the automata $\mathcal{A}_i$ is fed back to the input channel. Automaton $\mathcal{A}_1$ has the following (deterministic) behaviour: It outputs a token; waits for a token on input and then outputs another token. Automaton $\mathcal{A}_2$ has the choice between two behaviours: Either it outputs a token and stops, *or* it waits for an input token, then outputs two tokens. For both automata, the IO-relation relates empty input to zero or one output token, and non-empty input to zero, one or two output tokens. But inserted in the context as illustrated above, $\mathcal{A}_1$ can output two tokens, whereas $\mathcal{A}_2$ can only output a single token, choosing its first behaviour. This example shows very clearly that it is necessary to look for a model that records a more detailed causality relation than the IO-relations.

Jonsson [17] and Kok [25] have independently given fully abstract models for nondeterminate dataflow. Jonsson's model is based on trace[1] sets, which are sets of possible interactions between a process and its environment. Kok's model turned out to be equivalent. They showed that this model is fully abstract for indeterminate dataflow networks with a fair merge primitive, which was then shown by Russell [35] to hold even for dataflow networks with the weakest nondeterministic primitive, bounded choice[2]. Rabinovich and Traktenbrot analyzed the same issues from the point of view of finite observations and came up with general conditions under which a Kahn-like principle would hold [32, 33, 34].

3 A Traced Monoidal Category of Kahn Processes

In this section we summarize the basic theory of traced monoidal categories and then describe a category of Kahn processes as an instance of a traced monoidal category. The notion of traced monoidal category abstracts the notion of trace of a matrix from multilinear algebra. However it has emerged in a variety of new contexts including the study of feedback systems [3], knot theory [16] and recursion [15]. The axiomatization presented below is the definition of Joyal, Street and Verity [20] but specialized to the context of symmetric monoidal categories so that the axioms appear simpler; in particular we do not consider braiding or twists. In this paper the fact that trace models feedback (or iteration) is attributed to Bloom, but as far back as 25 years ago Bainbridge had been studying trace in the context of feedback in systems and control theory. Indeed Bainbridge had noticed that there were two kinds of trace (associated with two different monoidal structures) in $\mathbf{Rel}$ and that the powerset functor moves one between these situations. Furthermore he noted that one of the traces corresponds to feedback in what are essentially memoryless Kahn networks [3].

3.1 Traced Monoidal Categories

In this section we give the axioms for a symmetric monoidal category equipped with a trace. We assume that the reader is familiar with the notion of a symmetric tensor product. We write $\otimes$ for the tensor product and $\sigma_{XY} : X \otimes Y \longrightarrow Y \otimes X$ for the natural isomorphism (the symmetry) in this case.

[1]This word commonly used in the literature unfortunately clashes with "trace" in linear algebra. Normally this is not a problem but the present paper uses this word in both senses, we hope the reader will be able to disambiguate from context.

[2]See [30, 29] for a study of the differences of nondeterminate primitives.

[3]We are indebted to Samson Abramsky for pointing this reference out to us.

Definition 1 *A **trace** for a symmetric monoidal category $\mathcal{C}$ is family of functions*

$$Tr^U_{X,Y}() : \mathcal{C}(X \otimes U, Y \otimes U) \to \mathcal{C}(X, Y)$$

satisfying the following conditions

1. Naturality I*: Given* $f\colon Z \otimes U \longrightarrow Y \otimes U$ *and* $g\colon X \longrightarrow Z$
$$Tr^U_{X,Y}(f \circ (g \otimes I_U)) = Tr^U_{Z,Y}(f) \circ g.$$
2. Naturality II*: Given* $f\colon X \otimes U \longrightarrow Z \otimes U$ *and* $g\colon Z \longrightarrow Y$
$$Tr^U_{X,Y}((g \otimes I_U) \circ f) = g \circ Tr^U_{X,Z}(f)$$
3. Dinaturality*: Given* $f\colon X \otimes U \longrightarrow Y \otimes V$ *and* $g\colon V \longrightarrow U$
$$Tr^U_{X,Y}((I_Y \otimes g) \circ f) = Tr^V_{X,Y}(f \circ (I_X \otimes g)).$$
4. Bekic*:* $f\colon X \otimes U \otimes V \longrightarrow Y \otimes U \otimes V$
$$Tr^{U \otimes V}_{X,Y}(f) = Tr^U_{X,Y}(Tr^V_{X \otimes U, Y \otimes U}(f)).$$
5. Yanking*:* $Tr^U_{U,U}(\sigma_{UU}) = I_U$.
6. Superposing*: Given* $f\colon X \otimes U \longrightarrow Y \otimes U$ *and* $g\colon W \longrightarrow Z$
$$Tr^U_{X \otimes W, Y \otimes Z}((I_Y \otimes \sigma_{UZ}) \circ (f \otimes g) \circ (I_X \otimes \sigma_{WU})) = Tr^U_{X,Y}(f) \otimes g.$$

The following proposition is an easy consequence of the definitions. It shows how composition can be defined from trace and tensor.

Proposition 2 *Given* $g\colon U \longrightarrow Y$ *and* $f\colon X \longrightarrow U$ *we have*

$$Tr^U_{X,Y}(\sigma_{UY} \circ (f \otimes g)) = g \circ f.$$

This could be viewed as a generalization of the yanking condition.

3.2 The Kahn Category

The basic intuitions behind Kahn networks are, of course, due to Kahn [21] and the formal development of the subject is due to Kahn and McQueen [22]. The particular axiomatization presented here builds on the ideas of Stark [38] but using the formalism of traces presented in [29]. No originality is claimed for the trace model, it was Bengt Jonsson [17] who showed that traces form a fully abstract model of dataflow networks and there were several others with similar ideas at the time.

We have a fixed set $\mathcal{V}$ of *values* and a fixed set $\mathcal{P}$ of *ports*. An *event* is a triple $\langle a, i/o, v\rangle$ where $a \in \mathcal{P}$ and $v \in \mathcal{V}$. We say that $\langle a, v\rangle$ is the *label* of the event $\langle a, i/o, v\rangle$. An event of the form $\langle a, o, v\rangle$ is called an *output* event and one of the form $\langle a, i, v\rangle$ is called an *input* event. We consider sequences of these events. If α is a sequence of events we write $\alpha|_o$ for the sequence of *labels* of output events discarding the input events, similarly for $\alpha|_i$. We write $\alpha|_A$ for the sequence obtained by keeping only the events on the ports in A and $\alpha|_{A_o}$ for the sequence of labels of all output events on A. We extend these notations to sets of sequences. We use the notation $\alpha \leq \beta$ for the prefix order on sequences. We write $\mathcal{I}_A$ for the set Ax$\{i\}$x$\mathcal{V}$ of all input events on ports in A and similarly $\mathcal{O}_A$ for Ax$\{o\}$x$\mathcal{V}$. Finaly, we write $\mathcal{L}_A$ for the set Ax$\mathcal{V}$ of labels on ports in A.

Definition 3 *A* ***process*** *of* ***sort*** (A, B)*, where* $A, B \subseteq \mathcal{P}$ *is a prefix closed set of finite sequences over the alphabet* $\mathcal{I}_A \cup \mathcal{O}_B$*. The set of sequences, say* S*, satisfies the following closure properties,* α *and* β *are sequences of events:*

1. *If* $\alpha\langle b, o, v\rangle\langle a, i, u\rangle\beta \in S$ *then* $\alpha\langle a, i, u\rangle\langle b, o, v\rangle\beta \in S$.
2. *If* $\alpha\langle b, o, v\rangle\langle b', o, u\rangle\beta \in S$ *and if* $b \neq b'$ *then* $\alpha\langle b', o, u\rangle\langle b, o, v\rangle\beta \in S$.
3. *If* $\alpha\langle a, i, u\rangle\langle a', i, v\rangle\beta \in S$ *and if* $a \neq a'$ *then* $\alpha\langle a', i, v\rangle\langle a, i, u\rangle\beta \in S$.
4. *If* $\alpha \in S$ *then* $\alpha\langle a, i, v\rangle$ *for all* $a \in A$ *and* $v \in \mathcal{V}$.

We call A *the* ***input ports*** *and* B *the* ***output ports*** *of the process.*

The last condition above is called *receptivity*, a process could receive any data on its input port; unlike with synchronous processes. Receptivity is the basic reason why traces suffice to give a fully-abstract model for asynchronous processes; in calculi with synchronous communication one needs branching information.

The first three conditions express concurrency conditions on events occurring at different ports. Note an asymmetry in the first condition. If an output occurs before an input then it could also occur after the input instead. However, if an output occurs after an input then the pair of events cannot be permuted because the output event may be in response to the input. Furthermore we are assuming, again in (1), that the arrival of input does not disable already enabled output. In an earlier investigation [30] these were called *monotone* automata and it was shown that many common primitives, such as fair merge, timeouts, interrupts and polling cannot be expressed as monotone automata.

Given processes as sets of sequences we define *composition* as follows. We begin by defining the shuffle of two sets of sequences.

Definition 4 *Given two sets of sequences of events, say* S *of sort* (A, B) *and* S' *of sort* (A', B')*, with* $A \cap A' = \varnothing = B \cap B'$*, we define the set* $S \Delta S'$ *(read,* S ***shuffle*** S'*) as the set of all sequences* γ *of sort* $(A \cup A', B \cup B')$ *satisfying the following conditions*

1. *$\gamma|_{A\cup B} \in S$ and*

2. *$\gamma|_{A'\cup B'} \in S'$.*

We then define composition, by picking from the shuffle, the sequences having the right causal precedence of events on B and then discarding these, now "internal", events.

Definition 5 *Given processes $f\colon A \longrightarrow B$ and $g\colon B \longrightarrow C$ we define the composite of f and g by $f; g \triangleq S|_{A_i \cup C_o}$, where $S \subseteq f\Delta g$ (with ports renamed if necessary to avoid name clashes) is the the largest set s.t. for any $\delta \in S$*

1. *$\delta|_{B_o} = \delta|_{B_i}$,*

2. *$\forall \delta' \leq \delta . \delta'|_{B_i} \leq \delta'|_{B_o}$.*

Proposition 6 *The composite of two processes does yield a process, i.e. the closure conditions are satisfied. There is an identity process and composition is associative.*

Definition 7 *The category **Kahn** of Kahn processes has as objects finite subsets of $\mathcal{P}$ and as morphisms from A to B, processes with A as the input ports and B as the output ports. Composition of morphisms is defined by composition of processes as defined above.*

The following proposition has a routine proof but is important to note.

Proposition 8 *The following construction defines a monoidal structure. Given $f\colon A \longrightarrow B$ and $f'\colon A' \longrightarrow B'$ we define $f \otimes f'\colon A \uplus A' \longrightarrow B \uplus B'$ as $f\Delta g$.*

The trace construction is as follows. Given $f\colon X \uplus U \longrightarrow Y \uplus U$ we define $Tr^U_{X,Y}(f)\colon X \longrightarrow Y$ as the set of all γ such that there is a sequence $\delta \in f$ with

1. $\delta|_{X_i \cup Y_o} = \gamma$,

2. $\delta|_{U_o} = \delta|_{U_i}$ and

3. $\forall \delta' \leq \delta . \delta|_{U_i} \leq \delta|_{U_o}$.

Theorem 9 *With the structures given above, **Kahn** is a traced monoidal category.*

The generalized yanking property can be interpreted in this category as saying that composition can be obtained as a combination of parallel composition (that is, shuffling) and feedback. This is a well-known fact in dataflow folklore.

4 Generalising relations

Kahn processes are typical of the solutions to the problem of obtaining a compositional semantics for nondeterministic dataflow, as illustrated by the causal anomaly. A correct compositional semantics is got by keeping track of the causal dependency between events. In this section we will describe another solution that comes about as a natural extension of the IO-relation model.

First let us give a category of Kahn IO-relations. For a process S of sort (A, B) define its IO-relation $R_S \triangleq \{ (\alpha|_{A_i}, \alpha|_{B_o}) \mid \alpha \in S\} \subseteq \mathcal{L}_A^* \times \mathcal{L}_B^*$. Actually, this defines a relation $\overline{R}_S$ in $(\mathcal{L}_A^*/\overline{\sim})\times(\mathcal{L}_B^*/\overline{\sim})$ where $\mathcal{L}_A^*/\overline{\sim}$ consists of the *Mazurkiewicz traces* [28] (or the elements in the free partially commutative monoid [12]) of the trace language $(\mathcal{L}_A^*, \mathcal{L}_A, I_A)$ where $I_A \subseteq \mathcal{L}_A \times \mathcal{L}_A$ is the *independence* relation defined by $\langle a, u\rangle \ I_A \ \langle a', v\rangle$ iff $a \neq a'$. Recall that as in [42] the traces are equivalence classes of $\overline{\sim}$, the smallest equivalence relation such that $\alpha w' \, w \beta \overline{\sim} \alpha w \, w' \beta$ if $w \ I_A \ w'$. For $\alpha \in \mathcal{L}_A^*$, let $\overline{\alpha}$ denote its Mazurkiewicz trace. The traces can be partial ordered by $\overline{\alpha} \sqsubseteq \overline{\beta}$ iff $\exists \gamma . \overline{\alpha\gamma} = \overline{\beta}$ (see Ch.7 of [42]). Let ϵ_A (or just ϵ) denote the empty trace. In the following, we will let $\overline{\mathbf{A}}$ refer to the partial order category given by $\mathcal{L}_A^*/\overline{\sim}$ and the ordering $\sqsubseteq$ and refer to these categories as the *path categories*.[4]

Any IO-relation $\overline{R}_S$ for a Kahn process S is monotone and receptive, i.e. for $\overline{\alpha}, \overline{\beta} \in \mathcal{L}_A^*/\overline{\sim}$, if $\overline{\alpha} \sqsubseteq \overline{\beta}$ then $\overline{\alpha}\overline{R}_S \subseteq \overline{\beta}\overline{R}_S$ and for $w \in \mathcal{L}_A$, if $(\overline{\alpha}, \overline{\beta}) \in \overline{R}_S$ then $(\overline{\alpha w}, \overline{\beta}) \in \overline{R}_S$. Relations of this kind correspond to functors $\overline{\mathbf{A}} \times \overline{\mathbf{B}}^{op} \rightarrow \mathbf{2}$, where $\mathbf{2}$ is the category consisting of two objects 0 and 1 and only one non-identity arrow $0 \rightarrow 1$. Viewing the relations in this way the composition of $R \colon \overline{\mathbf{A}} \times \overline{\mathbf{B}}^{op} \rightarrow \mathbf{2}$ and $R' \colon \overline{\mathbf{B}} \times \overline{\mathbf{C}}^{op} \rightarrow \mathbf{2}$ can be written as

$$\overline{\alpha}(R; R')\overline{\gamma} = \bigvee_{\overline{\beta} \in \overline{\mathbf{B}}} \overline{\alpha} R \overline{\beta} \wedge \overline{\beta} R' \overline{\gamma}, \tag{1}$$

where we make use of the obvious join and meet operations on $\mathbf{2}$. Such relations in fact form the arrows of a traced symmetric monoidal category, but as illustrated by the example in section 2.1 it cannot possibly be used to give a compositional and correct treatment of feedback for indeterminate dataflow. We need to be able to express differences in causal dependencies between input and output. This is precisely what moving to the bicategory of *profunctors* $\mathbf{Prof}$ allows us to do.

4.1 Profunctors

Profunctors, (or bimodules, or distributors [5]) are a categorical generalisation of sets and relations. The objects of $\mathbf{Prof}$ are small categories and arrows are profunctors; profunctors are like receptive monotone relations but with the category $\mathbf{2}$ replaced by $\mathbf{Set}$.

[4]The traces can also be viewed as a specific kind of pomsets [31] and the path categories as a subcategory of the category of pomsets given in [19].

Definition 10 *Let $\mathbf{P}$ and $\mathbf{Q}$ be small categories. A profunctor $X\colon \mathbf{P} \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q}$ is a bifunctor $X\colon \mathbf{P} \times \mathbf{Q}^{\mathbf{op}} \longrightarrow \mathbf{Set}$. For p, q objects of respectively $\mathbf{P}$ and $\mathbf{Q}$ we will write X^p_q for the application $(Xp)q$ and similarly for morphisms.*

The tensor product is given by the categorical product on objects and set-theoretic product on arrows.

Definition 11 *Let $\mathbf{P}$,$\mathbf{P}'$ and $\mathbf{Q}$,$\mathbf{Q}'$ be small categories and $X\colon \mathbf{P} \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q}$, $Y\colon \mathbf{P}' \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q}'$ profunctors. Taking $\mathbf{P} \otimes \mathbf{P}' \triangleq \mathbf{P} \times \mathbf{P}'$ and $X \otimes Y \triangleq X \times Y\colon \mathbf{P} \otimes \mathbf{P}' \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q} \otimes \mathbf{Q}'$, so $(X \otimes Y)^{p,p'}_{q,q'} = X^p_q \times Y^{p'}_{q'}$, defines a symmetric monoidal structure on $\mathbf{Prof}$.*

The canonical choice of trace on $\mathbf{Prof}$ (cf. [20]) is to take the trace of a profunctor $X\colon \mathbf{P} \otimes \mathbf{U} \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q} \otimes \mathbf{U}$ to be given pointwise by the *coend*, so

$$Tr^U_{P,Q}(X)^p_q = \int^u X^{p,u}_{q,u}.$$

Composition[5] is given by

$$(Y;Z)^p_u = \int^q Y^p_q \times Z^q_u,$$

for $Y\colon \mathbf{P} \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{Q}$ and $Z\colon \mathbf{Q} \mathrel{-\!\!\!+\!\!\!\!\rightarrow} \mathbf{U}$. This generalises the expression for relational composition given by equation (1) earlier.

Since we are working with functors into $\mathbf{Set}$, the coend has an explicit definition. We have

$$\int^u X^{p,u}_{q,u} \cong \biguplus_{u \in \mathbf{U}} \{\, x \in X^{p,u}_{q,u} \}_{/\sim}, \tag{2}$$

where $\sim$ is the symmetric, transitive closure of the relation $\rightsquigarrow$ defined as follows. For $x \in X^{p,u}_{q,u}$ and $x' \in X^{p,u'}_{q,u'}$, let

$$x \rightsquigarrow x' \text{ if } \exists m\colon u \longrightarrow u' \text{ and } y \in X^{p,u}_{q,u'} \text{ s.t. } x \xleftarrow{X^{1_p,1_u}_{1_q,m}} \!\!\dashv y \xmapsto{X^{1_p,m}_{1_q,1_{u'}}} x'.$$

Like the closely related model of [3], this model doesn't give an operationally correct treatment of dataflow as it stands. Taking the trace as given by a coend suffers from defects similar to those encountered with relations; the coend is too generous in the behaviour it allows, permitting communications not beginning at the initial communication. However, as we will see shortly, profunctors contain the additional causal information that makes an alternative definition of trace possible, one which like coends can be expressed as a colimit but which this time agrees with the trace on Kahn processes.

[5]This defines composition only to within isomorphism, explaining why we get a *bi*category.

4.2 An operational reading

To see the connection with Kahn processes we examine the structure of profunctors more closely. First, we restrict attention to *pointwise rooted profunctors* between path categories, which intuitively correspond to those profunctors having a unique initial state.

Definition 12 *Let $X\colon \overline{\mathbf{A}} \to\!\!\!\!\!\!\mapsto \overline{\mathbf{B}}$. If $X\overline{\alpha}$ is a rooted presheaf (i.e. $X^{\overline{\alpha}}_{\epsilon}$ is the singleton set) for any $\overline{\alpha}$ object of $\overline{\mathbf{A}}$, we say that X is a* pointwise rooted profunctor. *We denote the single element belonging to X^{ϵ}_{ϵ} by r_X (the root of X).*

It is easy to check that rootedness is satisfied by identities and preserved by composition, thus forming a category, which we will refer to as $\mathbf{Prof}_{\perp}$, the category of *port profunctors*. We write $X\colon \overline{\mathbf{A}} \overset{\perp}{\to\!\!\!\!\!\!\mapsto} \overline{\mathbf{B}}$ when X is a profunctor in $\mathbf{Prof}_{\perp}$. The category $\mathbf{Prof}_{\perp}$ inherits the symmetric monoidal structure of $\mathbf{Prof}$. Explicitly we can define $\overline{\mathbf{A}} \otimes \overline{\mathbf{B}} = \overline{\mathbf{A} \uplus \mathbf{B}}$. The coend fails to preserve rootedness in general.

Similar to the construction in [41], we have an operational interpretation of port profunctors, which is a slight generalisation of the well-known construction of the *category of elements* of a presheaf.

Definition 13 *Let $X\colon \overline{\mathbf{A}} \overset{\perp}{\to\!\!\!\!\!\!\mapsto} \overline{\mathbf{B}}$ be a port profunctor. Define its (A,B)-port automaton $El_{(A,B)}(X) \triangleq (S, i, \longrightarrow, E)$, where S is the set of* states, *$i \in S$ is the* initial state, *E is the set of* events, *and $\longrightarrow \subseteq S{\times}E{\times}S$ is the* transition relation, *given by*

- $S = \{\big((\overline{\alpha},\overline{\beta}),x\big) \mid (\overline{\alpha},\overline{\beta})$ *is an object in* $\overline{\mathbf{A}} \times \overline{\mathbf{B}}^{op}$ & $x \in X^{\overline{\alpha}}_{\overline{\beta}}\}$
- $i = \big((\epsilon,\epsilon), r_X\big)$
- $E = \mathcal{I}_A \cup \mathcal{O}_B$
- $\big((\overline{\alpha},\overline{\beta}),x\big) \overset{\mathsf{i}\,a,v}{\longrightarrow} \big((\overline{\alpha\langle a,v\rangle},\overline{\beta}),y\big)$, *if* $X^{m}_{1_{\overline{\beta}}}x = y$, *for* $m\colon \overline{\alpha} \longrightarrow \overline{\alpha\langle a,v\rangle}$ *an arrow of* $\overline{\mathbf{A}}$.
- $\big((\overline{\alpha},\overline{\beta}),x\big) \overset{\mathsf{o}\,b,v}{\longrightarrow} \big((\overline{\alpha},\overline{\beta\langle b,v\rangle}),y\big)$, *if* $X^{1_{\overline{\alpha}}}_{m}y = x$, *for* $m\colon \overline{\beta} \longrightarrow \overline{\beta\langle b,v\rangle}$ *an arrow of* $\overline{\mathbf{B}}$.

Define $Seq(X)$ to be the set of finite sequences of events labelling finite sequences of transitions of $El_{(A,B)}(X)$ beginning at the initial state.

As an example, the automata of the profunctors modelling the behaviour of the two automata $\mathcal{A}_1$ and $\mathcal{A}_2$ from section 2.1 can be pictured as follows:

$$\mathcal{A}_1\colon \quad \begin{tikzcd} {} & \bullet \arrow[l, dotted, "\mathsf{i}"'] & \\ {} \arrow[u, dotted] & \bullet \arrow[l, dotted, "\mathsf{i}"'] \arrow[u, "\mathsf{o}"] & \bullet \arrow[l, "\mathsf{i}"'] \\ {} \arrow[u, dotted] & \bullet \arrow[l, dotted, "\mathsf{i}"'] \arrow[u, "\mathsf{o}"] & i \arrow[l, "\mathsf{i}"'] \arrow[u, "\mathsf{o}"] \end{tikzcd} \qquad \mathcal{A}_2\colon \quad \begin{tikzcd} {} & \bullet \arrow[l, dotted, "\mathsf{i}"'] & \\ {} \arrow[u, dotted] & \bullet \arrow[l, dotted] \arrow[u, "\mathsf{o}"] & \bullet \arrow[l, "\mathsf{i}"'] \\ {} \arrow[u, dotted] & \star \arrow[l, dotted] & \\ {} \arrow[u, dotted] & \bullet \arrow[l, dotted] \arrow[uu, bend right, "\mathsf{o}"'] \arrow[u, "\mathsf{o}"] & i \arrow[l, "\mathsf{i}"'] \arrow[uu, "\mathsf{o}"'] \end{tikzcd} ,$$

repeating the same pattern infinitely to the left.

Remarkably, the axioms of receptivity and monotonicity usually imposed on monotone port automata [30] and the usual commutativity axiom of asynchronous transition systems [4, 37] follow simply by functoriality for port automata of profunctors.

Proposition 14 *Let $X : \overline{\mathbf{A}} \mathrel{-\!\!\perp\!\!\to} \overline{\mathbf{B}}$ be a port profunctor. The following axioms hold for $El_{(A,B)}(X) = (S, i, \longrightarrow, E)$*

A1. *Receptivity:* $\forall e \in \mathcal{I}_A, s \in S\, \exists! s' \in S.\ s \xrightarrow{e} s'$,

A2. *Monotonicity:*
$$\begin{tikzcd}
& s \arrow[dl, "\mathsf{o}\,b{,}v"'] \arrow[dr, "\mathsf{i}\,a{,}v'"] & \\
s_1 \arrow[dr, dotted, "\mathsf{i}\,a{,}v'"'] & \Downarrow & s_2 \arrow[dl, dotted, "\mathsf{o}\,b{,}v"] \\
& u &
\end{tikzcd}$$

A3. *Commutativity:*
$$\begin{tikzcd}
& s \arrow[dl, "e_1"'] \arrow[dr, dotted, "e_2"] & \\
s_1 \arrow[rr, phantom, "\Rightarrow"] \arrow[dr, "e_2"'] & & s_2 \arrow[dl, dotted, "e_1"] \\
& u &
\end{tikzcd}\quad ,$$

for $e_1, e_2 \in \mathcal{I}_A$ *or* $e_1, e_2 \in \mathcal{O}_B$.

This immediately gives the following corollary.

Corollary 15 *Let $X : \overline{\mathbf{A}} \mathrel{-\!\!\perp\!\!\to} \overline{\mathbf{B}}$ port profunctor. Then, $Seq(X)$ is a Kahn process.*

The failure of the coend definition of feedback is illustrated by the example above: As expressed by equation (2), the coend "quantifies" over all states on the IO-diagonal. In the case of $\mathcal{A}_2$, this include the "bad" state $\star$, that cannot be reached by a path on which output preceeds input. The idea in the definition of the trace to come, is exactly to restrict this quantification to only the "good" states with the correct causal precedence.

4.3 A relational model of indeterminate dataflow

We will restrict the port profunctors to those for which the associated port automata satisfies an additional axiom

A4. *Stability:* $s \neq s_2$ &
$$\begin{tikzcd}
& s \arrow[dl, "\mathsf{i}\,a{,}v"'] & \\
s_1 \arrow[rr, phantom, "\Rightarrow!"] & & u \arrow[ul, dotted, "\mathsf{i}\,a'{,}v'"'] \arrow[dl, dotted, "\mathsf{i}\,a{,}v"] \\
& s_2 \arrow[ul, "\mathsf{i}\,a'{,}v'"] &
\end{tikzcd}$$
& $a \neq a'$.

which amounts to requiring that the profunctors (regarded as functors to sets) preserve pullbacks in their input arguments. It implies that any output event depend on a unique sequence of input events (up to commutativity). This property is satisfied by identities, and from the results in this section it follows that it is preserved under composition and tensor. Thus, the stable port profunctors between path categories form a monoidal sub category of $\mathbf{Prof}_\perp$, which we will denote by $\mathbf{SProf}_\perp$.

In the following we will switch freely between elements of a profunctor and states of the associated port automata, using $\longrightarrow$ as a relation between elements. Note that the relation $\rightsquigarrow$, defined when giving the explicit definition of the coend, can be seen as a relation between states of port automata, and understood as a chain of *communication* pairs of the form $\bullet \xrightarrow{\mathrm{o}\,w} \bullet \xrightarrow{\mathrm{i}\,w} \bullet$. Recall that $\sim$ was defined to be the symmetric, transitive closure of $\rightsquigarrow$. For profunctors in $\mathbf{SProf}_\perp$ we can prove a diamond property, which follows from the stability condition.

Lemma 16 *Let $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$ be a stable port profunctor. If $x_1 \rightsquigarrow x_3$ and $x_2 \rightsquigarrow x_3$, for x_1, x_2, x_3 elements of X, then there exists an element x_0 such that $x_0 \rightsquigarrow x_1$ and $x_0 \rightsquigarrow x_2$.*

By induction, this gives us an important corollary.

Corollary 17 *Let $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$ be a stable port profunctor. If $x \sim y$, for x, y elements of X, then there exists an element z such that $z \rightsquigarrow^* x$ and $z \rightsquigarrow^* y$.*

We are now ready to give the definition of the restricted trace. Note that the $\rightsquigarrow$ transitions by definition maintains the causal precedence of feedback; extending output first and then input correspondingly.

Definition 18 *For $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$, define $Tr^C_{A,B}(X)\colon \overline{\mathbf{A}} \xrightarrow{\perp} \overline{\mathbf{B}}$, the trace of X as follows. For objects α, β of resp. $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$, let*

$$Tr^C_{A,B}(X)^{\overline{\alpha}}_{\overline{\beta}} = \biguplus_{\overline{\gamma} \in \overline{\mathbf{C}}} \{ x \in X^{\overline{\alpha},\overline{\gamma}}_{\overline{\beta},\overline{\gamma}} \mid r_X (\xrightarrow{A,B} \cup \rightsquigarrow)^* x \}_{/\sim},$$

where $\xrightarrow{A,B} = \longrightarrow \cap S_X \times (\mathcal{I}_A \cup \mathcal{O}_B) \times S_X$. For arrows m, n of resp. $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$, define $Tr^C_{A,B}(X)^m_n$ to be the map sending $[x]_\sim$ to $[X^{m,1_{\overline{\gamma}}}_{n,1_{\overline{\gamma}}} x]_\sim$, for $x \in X^{\overline{\alpha},\overline{\gamma}}_{\overline{\beta},\overline{\gamma}}$.

It follows from functoriality of X that this indeed is a profunctor and a simple inspection shows that rootedness is preserved. Preservation of stability follows from corollary 17.

The trace has an equivalent definition, based on the standard construction of the subdivision category [27] which allows any coend to be expressed as a colimit. For a category $\mathbf{Q}$ its subdivision category $\mathbf{Q}^\natural$ is defined as follows. The objects of $\mathbf{Q}^\natural$ are all symbols $q^\natural$ and $f^\natural$ for q object in $\mathbf{Q}$ and f arrow in $\mathbf{Q}$. The arrows of $\mathbf{Q}^\natural$ are the identity arrows for these objects, plus for each arrow $f\colon q_0 \longrightarrow q_1$ in $\mathbf{Q}$ two arrows $f_i\colon f^\natural \longrightarrow q_i^\natural$, $i = 0, 1$. There are no non-trivial compositions. A profunctor $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$ defines a profunctor $X^\natural\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}}^\natural \xrightarrow{+} \overline{\mathbf{B}}$ by $(X^\natural)^{\overline{\alpha},\overline{\gamma}^\natural}_{\overline{\beta}} \triangleq X^{\overline{\alpha},\overline{\gamma}}_{\overline{\beta},\overline{\gamma}}$ and for $f\colon \overline{\gamma} \longrightarrow \overline{\delta}$ an arrow of $\overline{\mathbf{C}}$, $(X^\natural)^{\overline{\alpha},f^\natural}_{\overline{\beta}} \triangleq X^{\overline{\alpha},\overline{\gamma}}_{\overline{\beta},\overline{\delta}}$. On arrows f_0, f_1, let $(X^\natural)^{m,f_0}_n \triangleq X^{m,1_{\overline{\gamma}}}_{n,f}$ and $(X^\natural)^{m,f_1}_n \triangleq X^{m,f}_{n,1_{\overline{\delta}}}$. It is a standard fact that

$$\int^{\overline{\gamma} \in \overline{\mathbf{C}}} X \cong \mathrm{Colim}_{\overline{\mathbf{C}}^\natural} X^\natural.$$

Now comes the non-standard part, restricting $X^\natural$ according to definition 18.

Definition 19 *For $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$, define $Fb^C_{A,B}(X)\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}}^\natural \xrightarrow{+} \overline{\mathbf{B}}$ as follows. For $\overline{\alpha}$, $\overline{\beta}$ and c objects of resp. $\overline{\mathbf{A}}$, $\overline{\mathbf{B}}$ and $\overline{\mathbf{C}}^\natural$, let*

$$Fb^C_{A,B}(X)^{\overline{\alpha},c}_{\overline{\beta}} = \{\, x \in (X^\natural)^{\overline{\alpha},c}_{\overline{\beta}} \mid r_X(\overset{A,B\otimes C}{\longrightarrow} \cup \rightsquigarrow)^* x \},$$

where $\overset{A,B\otimes C}{\longrightarrow} = \longrightarrow\ \cap S_X \times (\mathcal{I}_A \cup \mathcal{O}_{B\otimes C}) \times S_X$. For arrows let $Fb^C_{A,B}(X)^{m,f_i}_n \triangleq (X^\natural)^{m,f_i}_n|_{Fb^C_{A,B}(X)^{\overline{\alpha},f^\natural}_{\overline{\beta}}}$.

Actually, $Fb(-)$ extends to a functor between presheaf categories. The trace given in definition 18 can be expressed as a colimit as follows.

Proposition 20 *Let $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$ and $Fb^C_{A,B}(X)$ be given as above. Then,*

$$Tr^C_{A,B}(X) \cong \mathrm{Colim}_{\overline{\mathbf{C}}^\natural} Fb^C_{A,B}(X).$$

The following propositions are the key ingredients in showing that this indeed makes $\mathbf{SProf}_\perp$ a traced monoidal bicategory. First of all, one gets the usual profunctor composition from the trace.

Proposition 21 *Let $X : \overline{\mathbf{A}} \xrightarrow{\perp} \overline{\mathbf{B}}$ and $Y\colon \overline{\mathbf{B}} \xrightarrow{\perp} \overline{\mathbf{C}}$. Then,*

$$Tr^B_{A,C}(X \otimes Y) \cong \int^{\overline{\beta}} X_{\overline{\beta}} \otimes Y^{\overline{\beta}}.$$

Next, trace distributes through tensor and simultaneous trace is equivalent to iterated trace.

Proposition 22 *Let $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}}$ and $Y\colon \overline{\mathbf{A}'} \xrightarrow{\perp} \overline{\mathbf{B}'}$. Then,*

$$Y \otimes Tr^C_{A,B}(X) \cong Tr^C_{A'\otimes A, B'\otimes B}(Y \otimes X).$$

Proposition 23 *Let $X\colon \overline{\mathbf{A}} \otimes \overline{\mathbf{C}} \otimes \overline{\mathbf{D}} \xrightarrow{\perp} \overline{\mathbf{B}} \otimes \overline{\mathbf{C}} \otimes \overline{\mathbf{D}}$. Then,*

$$Tr^C_{A,B}(Tr^D_{A\otimes C, B\otimes C}(X)) \cong Tr^{C\otimes D}_{A,B}(X).$$

The proof of the latter proposition is clearly the most involved.

Theorem 24 *With the tensor structure and the trace operator given above, $\mathbf{SProf}_\perp$ is a traced monoidal category.*

As advertised, the trace indeed gives us the correct observational definition of feedback, proven by the existence of a functor from $\mathbf{SProf}_\perp$ to $\mathbf{Kahn}$, preserving the traced monoidal structure.

Proposition 25 *The map Seq of definition 13 defines the action on arrows of a traced monoidal functor $Seq\colon \mathbf{SProf}_\perp \longrightarrow \mathbf{Kahn}$, on objects simply mapping path categories to their underlying port set.*

5 Some consequences

We will briefly go through some of the consequences of having this categorical model of dataflow.

5.1 Bisimulation

The presentation of models for concurrency as categories allows us to apply a general notion of bisimulation from spans of open maps proposed in [19]. The general idea is to identify a *path category* $\mathbf{P} \hookrightarrow \mathbf{M}$ as a subcategory of the model $\mathbf{M}$, with objects representing runs or histories and morphisms compatible extensions of these. For a presheaf model $\hat{\mathbf{P}}$ the canonical choice is the category $\mathbf{P}$ under the yoneda embedding $\mathbf{y}$. Identifying the objects of $\mathbf{P}$ with their presheaf under $\mathbf{y}$, the notion of *open maps* specialized to presheaves is defined as follows.

Definition 26 *Let X, Y be objects of $\hat{\mathbf{P}}$ and $f\colon X \longrightarrow Y$ a morphism. Then f is* $\mathbf{P}$-open *if whenever for two path objects P, Q of $\mathbf{P}$ and morphism m, p, q such that the diagram commutes, there exists a morphism $h\colon Q \longrightarrow X$ as indicated by the dotted line, making the two triangles commute.*

$$\begin{tikzcd}
P \arrow[r, "p"] \arrow[d, "m"'] & X \arrow[d, "f"] \\
Q \arrow[r, "q"] \arrow[ur, dotted, "h"] & Y
\end{tikzcd}$$

Two objects X, Y of $\hat{\mathbf{P}}$ are said to be *P-bisimilar* iff there exists a *span of open maps* f_1, f_2:

$$X \xleftarrow{f_1} Z \xrightarrow{f_2} Y.$$

This gives a notion of bisimulation for profunctors between $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$; recall that $X\colon \overline{\mathbf{A}} \dashrightarrow \overline{\mathbf{B}}$ can be viewed as a presheaf in $\widehat{\overline{\mathbf{A}}^{op} \times \overline{\mathbf{B}}}$ where $\overline{\mathbf{A}}^{op} \times \overline{\mathbf{B}}$ is the canonical choice of path category. Viewed as a port-automaton, a path-object $(\overline{\alpha}, \overline{\beta})$ looks like

$$i \xrightarrow{\mathsf{i}\,\alpha_0} \bullet \xrightarrow{\mathsf{i}\,\alpha_1} \cdots \xrightarrow{\mathsf{i}\,\alpha_n} \bullet \xrightarrow{\mathsf{o}\,\beta_0} \bullet \xrightarrow{\mathsf{o}\,\beta_1} \cdots \xrightarrow{\mathsf{o}\,\beta_m} x,$$

closed under axioms A1-A3. As in [41], the bisimulation can be characterised as a back&forth bisimulation between the states of the associated port automata.

It is important to check that bisimulation on $\mathbf{SProf}_\perp$ is a congruence with respect to the operations tensor and trace. Here we can exploit some general properties of open maps and so bisimulation on presheaves: the product of (surjective) open maps in a presheaf category is (surjective) open [18]; any colimit-preserving functor between presheaf categories preserves (surjective) open maps [8]. The proof that trace on $\mathbf{SProf}_\perp$ preserves bisimulation uses the latter property, exploiting the fact that trace can be expressed as a colimit; first showing from the definition that $Fb(-)$ preserves open maps. The proof of the corresponding result for tensor rests on a construction of tensor from more basic functors. The tensor of $X_1 \colon \mathbf{P}_1 \mathrel{\mapstochar\mkern-3mu\rightarrow} \mathbf{Q}_1$ and $X_2 \colon \mathbf{P}_2 \mathrel{\mapstochar\mkern-3mu\rightarrow} \mathbf{Q}_2$ can be expressed as a product of presheaves over $\mathbf{P}_1^{op} \times \mathbf{Q}_1 \times \mathbf{P}_2^{op} \times \mathbf{Q}_2$:

$$X_1 \otimes X_2 = (\pi_1^* X_1) \times (\pi_2^* X_2)$$

where e.g.

$$\pi_1^* \colon \widehat{\mathbf{P}_1^{op} \times \mathbf{Q}_1} \to \mathbf{P}_1^{op} \times \widehat{\mathbf{Q}_1 \times \mathbf{P}_2^{op}} \times \mathbf{Q}_2$$

is obtained by composition with the projection

$$\pi_1 \colon \mathbf{P}_1^{op} \times \mathbf{Q}_1 \times \mathbf{P}_2^{op} \times \mathbf{Q}_2 \to \mathbf{P}_1^{op} \times \mathbf{Q}_1,$$

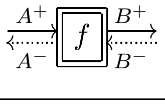
so $(\pi_1^* X_1)^{p_1,p_2}_{q_1,q_2} = {X_1}^{p_1}_{q_1}$. For general reasons π_1^* has a right adjoint (constructed as a right Kan extension—see [27, 19]). Thus π_1^* and, similarly, π_2^* are left adjoints and so preserve (surjective) open maps. Combined with the similar fact about product of presheaves we deduce that $\otimes$ preserves (surjective) open maps, and so bisimulation.

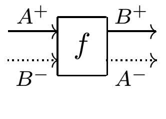
5.2 Higher types via Geometry of Interaction

The geometry of interaction programme can be seen as a method of constructing a compact closed category from a traced monoidal category.[6] As such it gives a method for realizing higher-order constructs in terms of feedback. In our setting one takes the categories $\mathbf{Kahn}$ and $\mathbf{SProf}_\perp$ and constructs compact-closed categories $\mathbf{HKahn}$ and $\mathbf{HProf}_\perp$ which then serve as the interpretations of higher-order Kahn processes and Port profunctors.

In this section we give a summary of a categorical presentation of the geometry of interaction construction due to Abramsky [1] and also to Joyal, Street and Verity [20]. We do not need the full generality of the latter presentation since we do not consider braiding or twists. Essentially, one obtain a higher-order model by working with processes with bi-directional "input" and "output". For dataflow this can be understood as splitting channels into a positive part and a negative part; the positive channels carry tokens in the usual direction and the negative in the opposite direction. These processes are implemented by uni-directional processes of the underlying category in the obvious way, regarding negative input channels as output channels and negative output as input. Below we will use box-diagrams in the style of [20], using double boxes for morphisms of the higher-order category.

Definition 27 *Given a traced monoidal category $\mathcal{C}$ we define a new category $\mathcal{G}(\mathcal{C})$ as follows. The objects of $\mathcal{G}(\mathcal{C})$ are pairs of objects (A^+, A^-) of $\mathcal{C}$. A morphism $f : (A^+, A^-) \longrightarrow (B^+, B^-)$ of $\mathcal{G}(\mathcal{C})$ is a $\mathcal{C}$-morphism $f : A^+ \otimes B^- \longrightarrow B^+ \otimes A^-$, ie.*



is implemented by


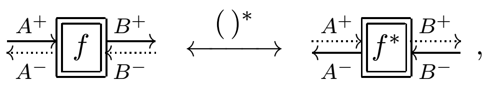
,

[6] In Girard's original treatment this was expressed in terms of traces in the category of Hilbert spaces. That situation is more complicated because not every morphism has a trace, so the categorical presentation of geometry of interaction is not the same as the original program.

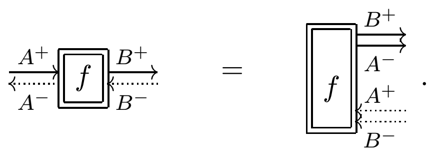
where dotted lines indicate channels that play the opposite role in $\mathcal{G}(\mathcal{C})$. Composition is implemented using composition, trace and symmetries of $\mathcal{C}$ to connect B-channels with same polarity, ie. for $g : (B^+, B^-) \longrightarrow (C^+, C^-)$, $f; g$ is implemented by $Tr^{B^-}_{A^+\otimes C^-, C^+\otimes A^-}(\sigma; (f \otimes I_{C^-}); \sigma'; (g \otimes I_{A^-}); \sigma'')$, for the appropriate symmetries σ, σ' and σ''.

Note that $\mathcal{C}$ embeds into $\mathcal{G}(\mathcal{C})$ as arrows with no negative flow, mapping objects A to (A, I).

A symmetric monoidal structure $\odot$ is defined on objects by $(A^+, A^-)\odot(B^+, B^-) = (A^+ \otimes B^+, B^- \otimes A^-)$ and for arrows f, g, we define $f \odot g$ by $\sigma; f \otimes g; \sigma'$, where σ, σ' are symmetry morphisms of $\mathcal{C}$, gathering channels of the same polarity. The symmetry morphism for $\odot$ is the evident tensor of symmetry morphisms of $\mathcal{C}$. We have an obvious duality defined on objects by $(A^+, A^-)^* = (A^-, A^+)$, and on arrows by swapping the roles of channels



defining a contravariant functor $(-)^* : \mathcal{G}(\mathcal{C}) \longrightarrow \mathcal{G}(\mathcal{C})$. Finally, from definition 27 we get internal hom sets by $(A^+, A^-) \multimap (B^+, B^-) = (B^+, B^-) \odot (A^+, A^-)^*$ corresponding to moving all channels to the output side, and changing their roles accordingly



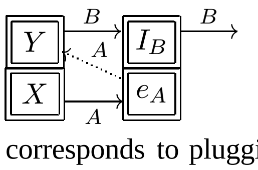
This defines a compact closed structure [24].

Proposition 28 *The category $\mathcal{G}(\mathcal{C})$ is a compact-closed category.*

We immediately get, since it preserves tensor and trace, that the functor $Seq\colon \mathbf{SProf}_\perp \longrightarrow \mathbf{Kahn}$ extends to one between the higher-order categories.

Proposition 29 *We have a functor $HSeq\colon \mathbf{HProf}_\perp \longrightarrow \mathbf{HKahn}$, defined using Seq on the base category.*

Within $\mathbf{HKahn}$ and $\mathbf{HProf}_\perp$, the diagonal $d_X\colon I \longrightarrow X \otimes X^*$ and evaluation $e_X\colon X^* \otimes X \longrightarrow I$ maps are essentially "routers", copying values from in-going channels to the corresponding out-going ones.



Indeed, for $X\colon I \longrightarrow A$ and $Y\colon A \longrightarrow B$ of say $\mathbf{SProf}_\perp$, when imbedded into $\mathbf{HProf}_\perp$ and internalized we get $(Y \otimes X)\colon I \longrightarrow (B \otimes A, A)$ as illustrated to the left, and the application $(Y \otimes X); (I_B \otimes e_A)\colon I \longrightarrow B$ simply corresponds to plugging X into Y in $\mathbf{SProf}_\perp$. The same goes for higher-order evaluation.

Summa summarum: We have two genuine higher-order calculi of dataflow, explicitly related to each other and based on a low-level implementation such that application simply corresponds to plugging networks together, combining wires using feedback.

6 Concluding remarks

It remains to systematically explore the full family of models for dataflow, relating automata, event structure and traces-based models to the relational model, following the pattern set in [42]. Work is underway on a bicategory of port automata to close a gap in [36]. This will provide further operational back up to the trace on port profunctors and help in the understanding of independence at higher-order. The higher-order models should be compared to the related, but clearly different, work in [2]. It remains to incorporate fairness into the profunctor model; it is hoped to exploit independence along the lines in [9]. A compositional semantics of Verilog, or perhaps an interesting fragment, feels within reach.

References

[1] ABRAMSKY, S. Retracing some paths in process algebra. In *CONCUR'96*, vol. 1119 of *LNCS*, pp. 1–17.

[2] ABRAMSKY, S., GAY, S., AND NAGARAJAN, R. Interaction categories and the foundations of typed concurrent programming. In *Proc. of the 1994 Marktoberdorf summer school*, Springer.

[3] BAINBRIDGE, E. S. Feedback and generalized logic. *Information and Control*, 31 (1976), 75–96.

[4] BEDNARCZYK, M. A. *Categories of asynchronous systems*. PhD thesis, University of Sussex, 1988.

[5] BORCEUX, F. *Handbook of categorical logic*, vol. 1. Cambridge University Press, 1994.

[6] BROCK, J., AND ACKERMAN, W. Scenarios: a model of non-determinate computation. In *Formalization of Programming Concepts* (1981), Diaz and Ramos, Eds., vol. 107 of *LNCS*, Springer.

[7] CATTANI, G. L., STARK, I., AND WINSKEL, G. Presheaf models for the pi-calculu. In *CTCS'97*, vol. 1290 of *LNCS*, Springer, pp. 106–126.

[8] CATTANI, G. L., AND WINSKEL, G. Presheaf models for concurrency. In *CSL'96*, vol. 1258 of *LNCS*, Springer, pp. 58–75.

[9] CHENG, A. Petri nets, traces, and local model checking. Tech. Rep. RS-95-39, BRICS, 1995.

[10] DENNIS, J. First version of a dataflow procedure language. In *Proc Colloque sur la Programmation* (1974), B. Robinet, Ed., vol. 19 of *LNCS*, Springer, pp. 362–376.

[11] DENNIS, J. B. *Control Flow and Data Flow: Concepts of Distributed Programming*, vol. 14 of *NATO ASI Series F*. Springer, 1984, ch. Data Flow Computation, pp. 345–398.

[12] DIEKERT, V., AND MÉTIVIER, Y. *Handbook of Formal Languages.* Springer, 1997, ch. Partial Commutation and Traces.

[13] FAUSTINI, A. A. An operational semantics for pure dataflow. In *ICALP'82*, vol. 140 of *LNCS*, Springer, pp. 212–224.

[14] GORDON, M. The semantic challenge of verilog hdl. www.cl.cam.ac.uk/mjcg/, April 1996. Revised version of an invited paper published in LICS'95.

[15] HASEGAWA, M. Recursion from cyclic sharing: traced monoidal categories and models of cyclic lambda calculi. In *TLCA'97* (1997), vol. 1210 of *LNCS*, pp. 196–213.

[16] JONES, V. F. A polynomial invariant for links via von neumann algebras. *Bull. Amer. Math. Soc. 129* (1985), 103–112.

[17] JONSSON, B. A fully abstract trace model for dataflow networks. In *POPL'89*, ACM, pp. 155–165.

[18] JOYAL, A., AND MOERDIJK, I. A completeness theorem for open maps. *Annals of Pure and Applied Logic 70*, 1 (1994), 51–86.

[19] JOYAL, A., NIELSEN, M., AND WINSKEL, G. Bisimulation from open maps. Tech. Rep. RS-94-7, BRICS, 1994.

[20] JOYAL, A., STREET, R., AND VERITY, D. Traced monoidal categories. vol. 119 of *Math. Proc. Camb. Phil. Soc.*, pp. 447–468.

[21] KAHN, G. The semantics of a simple language for parallel programming. In *Information Processeing* (1974), vol. 74, pp. 471–475.

[22] KAHN, G., AND MACQUEEN, D. Coroutines and networks of parallel processes. In *Proceedings of Information Processing* (1977), Gilchrist, Ed., North-Holland, pp. 993–998.

[23] KATIS, P., SABADINI, N., AND WALTERS, R. Bicategories of processes. *Journal of Pure and Applied Algebra* (1997).

[24] KELLY, G., AND LAPLAZA, M. Coherence for compact closed categories. *Journal of Pure and Applied Algebra 19* (1980), 193–213.

[25] KOK, J. A fully abstract semantics for dataflow nets. In *Proceedings of Parallel Architectures And Languages Europe* (Berlin, 1987), Springer, pp. 351–368.

[26] LYNCH, N. A., AND STARK, E. W. A proof of the kahn principle for input/output automata. *Information and Computation 82* (1989), 81–92.

[27] MAC LANE, S. *Categories for the Working Mathematician*. Springer, 1971.

[28] MAZURKIEWICZ, A. Trace theory. In *Petri Nets: Applications and Relationships to Other Models of Concurrency* (1986), vol. 255 of *LNCS*, Springer, pp. 279–324.

[29] PANANGADEN, P., AND SHANBHOGUE, V. The expressive power of indeterminate dataflow primitive. *Information and Computation 98*, 1 (1992), 99–131.

[30] PANANGADEN, P., AND STARK, E. W. Computations, residuals and the power of indeterminacy. In *Proc. of the 15th ICALP* (1988), Springer, pp. 439–454.

[31] PRATT, V. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 1 (1986).

[32] RABINOVICH, A., AND TRAKHTENBROT, B. A. Nets of processes and dataflow. To appear in Proceedings of ReX School on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, LNCS, 1988.

[33] RABINOVICH, A., AND TRAKHTENBROT, B. A. Nets and data flow interpreters. In *Proceedings of the 4th LICS* (1989), pp. 164–174.

[34] RABINOVICH, A., AND TRAKHTENBROT, B. A. Communication among relations. In *Proc. of the 7th ICALP* (1990), M. S. Paterson, Ed., vol. 443 of *LNCS*, Springer, pp. 294–307.

[35] RUSSELL, J. R. Full abstraction for nondeterministic dataflow networks. In *FOCS'89*, pp. 170–176.

[36] SELINGER, P. First-order axioms for asynchrony. In *CONCUR'97*, vol. 1243 of *LNCS*, Springer, pp. 376–390.

[37] SHIELDS, M. W. Concurrent machines. *Computer Journal 28* (1985), 449–465.

[38] STARK, E. W. Compositional relational semantics for indeterminate dataflow networks. In *CTCS* (Manchester, U.K., 1989), vol. 389 of *LNCS*, Springer, pp. 52–74.

[39] STARK, E. W. Dataflow networks are fibrations. Tech. rep., Dept. of Computer Science, Stony Brook, 1989.

[40] WINSKEL, G. A presheaf semantics of value-passing processes. In *CONCUR'96*, vol. 1119 of *LNCS*, pp. 98–114.

[41] WINSKEL, G., AND NIELSEN, M. Presheaves as transition systems. In *POMIV'96*, vol. 29.

[42] WINSKEL, G., AND NIELSEN, M. *Handbook of Logic in Computer Science*, vol. IV. OUP, 1995, ch. Models for concurrency.