Logical Methods in Computer Science
Volume 19, Issue 1, 2023, pp. 2:1–2:24
https://lmcs.episciences.org/

Submitted Mar. 11, 2022
Published Jan. 12, 2023

STREAM PROCESSORS AND COMODELS

RICHARD GARNER

School of Mathematical and Physical Sciences, Macquarie University, NSW 2109, Australia
e-mail address: richard.garner@mq.edu.au

Abstract. In 2009, Hancock, Pattinson and Ghani gave a coalgebraic characterisation of stream processors $A^{\mathbb{N}} \to B^{\mathbb{N}}$ drawing on ideas of Brouwerian constructivism. Their stream processors have an *intensional* character; in this paper, we give a corresponding coalgebraic characterisation of *extensional* stream processors, i.e., the set of continuous functions $A^{\mathbb{N}} \to B^{\mathbb{N}}$. Our account sites both our result and that of *op. cit.* within the apparatus of *comodels* for algebraic effects originating with Power–Shkaravska. Within this apparatus, the distinction between intensional and extensional equivalence for stream processors arises in the same way as the the distinction between *bisimulation* and *trace equivalence* for labelled transition systems and probabilistic generative systems.

1. Introduction

As is well known, the type of infinite *streams* of elements of some type A may be defined to be the final coalgebra $\nu X.\, A \times X$. If types are mere sets, then this coalgebra is manifested as the set $A^{\mathbb{N}}$ of infinite lists of A-elements, with the structure map

$$\alpha \colon \vec{a} \mapsto (a_0, \partial\vec{a}) \qquad \text{where} \qquad \partial(a_0, a_1, a_2, \dots,) = (a_1, a_2, \dots)\ . \tag{1.1}$$

Of course, the coalgebra structure describes the corecursive nature of streams, but also captures their sequentiality: an A-stream is *first* an A-value, and *then* an A-stream.

If A and B are types, then an *A-B-stream processor* is a way of turning an A-stream into a B-stream. If types are sets, then the crudest kind of stream processor would simply be a function $f \colon A^{\mathbb{N}} \to B^{\mathbb{N}}$; however, it is more computationally reasonable to restrict to those f which are *productive*, in the sense that determining each B-token of the output should require examining only a finite number of A-tokens of the input.

The productive functions $f \colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ are in fact precisely the *continuous* ones for the prodiscrete (= Baire) topologies on $A^{\mathbb{N}}$ and $B^{\mathbb{N}}$. While this representation of stream processors is mathematically smooth, it fails to make explicit their sequentiality: we should like to see the fact that determining each *successive* token of the output B-stream requires examining *successive* finite segments of the input A-stream. Much as for streams themselves, this can be done by presenting stream processors as a final coalgebra.

Key words and phrases: Comodels, residual comodels, bimodels, streams, stream processors, trace.
Supported by ARC grants FT160100393 and DP190102432.

Such a presentation was given in [HPG09]. Therein, the *type of A-B-stream processors* was taken to be the final coalgebra $\nu X.\, T_A(B \times X)$, where $T_A(V) = \mu X.\, V + X^A$; and it was explained how each element of this type encodes a continuous function $A^{\mathbb{N}} \to B^{\mathbb{N}}$, and how, conversely, each such continuous function yields an element of this type. An interesting aspect of the story is that these assignments are *not* mutually inverse: distinct elements of $\nu X.\, T_A(B \times X)$ may represent the same continuous function, so that elements of this type are really *intensional* representations of stream-processing algorithms.

While there are many perspectives from which this is a good thing, it leaves open the question of whether there is a coalgebraic representation for *extensional* stream processors, i.e., for the set of continuous functions $A^{\mathbb{N}} \to B^{\mathbb{N}}$. In this paper, we show that there is:

Theorem 1.1. *Let A and B be sets. The set of continuous functions $A^{\mathbb{N}} \to B^{\mathbb{N}}$ is the underlying set of the terminal B-ary comagma in the category of A-ary magmas.*

In this result, an *A-ary magma* is a set X with an operation $\xi \colon X^A \to X$ satisfying no further axioms. More generally, we can speak of A-ary magmas in any category $\mathcal{C}$ with products; while, if $\mathcal{C}$ is a category with *co*products, we can define an A-ary *co*magma in $\mathcal{C}$ to be an A-ary magma in $\mathcal{C}^{\mathrm{op}}$. Explicitly, this involves an object $X \in \mathcal{C}$ and a map $X \to X + \cdots + X$ into the coproduct of A copies of X, subject to no further conditions.

On the face of it, our Theorem 1.1 has no obvious relation to [HPG09], nor to anything resembling computation. Thus, the broader contribution of this paper is to site both the ideas of [HPG09] and our Theorem 1.1 within the well-established machinery of *comodels* [PS04, PP08], as we now explain.

The category-theoretic approach to computational effects originates in [Mog91]: given a monad T on a category of types and programs, we view elements of $T(V)$ as computations with side-effects from T returning values in V. This idea was refined in [PP02]; rather than considering monads *simpliciter*, we generate them from *algebraic theories* whose basic operations are the computational primitives for the effects at issue. A key example, for us, is the theory $\mathbb{T}_A$ of input from an alphabet A, which is freely generated by a single A-ary operation read.

The approach via algebraic theories has the virtue of giving a good notion of *model* in any category with finite powers. In particular, one has *comodels*, which are models in the opposite of the category of sets, and a key insight of [PS04] is that comodels of a theory $\mathbb{T}$ can be seen as coalgebraic objects for evaluating $\mathbb{T}$-computations to values. We recall these developments in detail in Section 2, and in particular, we see that a $\mathbb{T}_A$-comodel is an $A \times (-)$-coalgebra, and that the *final* comodel is the set of A-streams.

A range of authors [PP08, MS14, PS15, PS16, Uus15, KRU20, AB20, GMS20, UV20, Gar21] have taken this attractive perspective on operational semantics further. Particularly salient for us is the concept, due to [AB20, KRU20, UV20] of a *residual comodel.* Given theories $\mathbb{T}$ and $\mathbb{R}$, an $\mathbb{R}$-residual $\mathbb{T}$-comodel is, formally speaking, a comodel of $\mathbb{T}$ in the Kleisli category of $\mathbb{R}$ but is, practically speaking, a coalgebraic entity for evaluating or compiling $\mathbb{T}$-computations into $\mathbb{R}$-computations. In particular, we have $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodels, which translate requests for B-input into requests for A-input, and a little thought shows that this is exactly the rôle filled by an A-B-stream processor. In fact, the final coalgebra of [HPG09] turns out to be precisely the *final $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel*; in Sections 3 and 4, we explain this, and show how other aspects of [HPG09] such as the *composition* of intensional stream processors flow naturally.

To get from here to our Theorem 1.1 requires a new import from category-theoretic universal algebra: the notion of a *bimodel* [Fre66, TW70, BH96]. An $\mathbb{R}$-$\mathbb{T}$-*bimodel* is a comodel of $\mathbb{T}$ in the category of *Set*-models of $\mathbb{R}$. Since this latter category has the Kleisli category as a full subcategory, bimodels are a generalisation of residual comodels—one which, roughly speaking, allows additional quotients to be taken. These quotients are just what one needs to collapse the intensional stream processors of [HPG09] to their underlying continuous functions. We develop this theory in Section 5, culminating in our Theorem 1.1 which we now recognise as describing the *final* $\mathbb{T}_A$-$\mathbb{T}_B$-*bimodel*.

An obvious question at this point is whether we have similar characterisations of the final bimodel on replacing $\mathbb{T}_A$ and $\mathbb{T}_B$ by more elaborate algebraic theories. One step in this direction is given in [Yos22], which characterises the final $\mathbb{R}$-$\mathbb{T}$-bimodel whenever $\mathbb{R}$ and $\mathbb{T}$ are *free* algebraic theories, i.e., theories generated by operations subject to no equations.

A different direction of generalisation points towards labelled transition systems and generative probabilistic systems [vGSST90]. Indeed, (non-terminating, finitely branching) labelled transition systems with alphabet A are precisely $\mathbb{P}_f^+$-residual $\mathbb{T}_A$-comodels, for $\mathbb{P}_f^+$ the theory of binary non-deterministic choice; while (finitely supported) generative probabilistic systems with alphabet A are $\mathbb{D}$-residual $\mathbb{T}_A$-comodels, for $\mathbb{D}$ the theory of binary probabilistic choice. As is well known, in these examples, the final residual $\mathbb{T}_A$-comodel captures states up to *bisimulation* equivalence. What is perhaps less well known is that the final *bimodel* captures states up to *trace* equivalence; indeed, as shown in [Gar18, §7], the final $\mathbb{P}_f^+$-$\mathbb{T}_A$-bimodel is the set of closed subsets of $A^{\mathbb{N}}$; while the final $\mathbb{D}$-$\mathbb{T}_A$-bimodel is the set of probability measures on $A^{\mathbb{N}}$. This fact provides an alternative perspective on the trace semantics of [HJS07] (which itself builds on [PT99]) in which an object of traces is found as a final object among not all bimodels, but among all *free* bimodels; in future work, we will give a more careful comparison of the two notions of trace.

From this perspective, then, the continuous function encoded by an intensional stream processor can be seen as its "trace", and with this in mind, the final contribution of this paper in Section 6 is to explain from a comodel-theoretic perspective the fact that intensional stream processors admit a procedure of "normalisation-by-trace-evaluation", which normalises each intensional stream processor to a *maximally lazy* stream processor with the same trace; this is a particular instantiation of a more general schema which is explored further in [Gar18].

2. Streams as a final comodel

In this background section, we recall how algebraic theories present notions of effectful computation, how *comodels* of a theory furnish environments appropriate for evaluating such computations, and how the type of streams arises as a final comodel.

Definition 2.1 (Algebraic theory)**.** A *signature* comprises a set Σ of *function symbols*, and for each $\sigma \in \Sigma$ a set $|\sigma|$, its *arity*. Given a signature Σ and a set V, we define the set $\Sigma(V)$ of Σ-*terms with variables in* V by the inductive clauses

$$v \in V \implies v \in \Sigma(V) \quad \text{and} \quad \sigma \in \Sigma,\, t \in \Sigma(V)^{|\sigma|} \implies \sigma(t) \in \Sigma(V)\ .$$

An *equation* over a signature Σ is a formal equality $t = u$ between terms in the same set of free variables. A (algebraic) *theory* $\mathbb{T}$ comprises a signature and a set $\mathcal{E}$ of equations over it.

Definition 2.2 ($\mathbb{T}$-terms)**.** Given a signature Σ and terms $t \in \Sigma(V)$ and $u \in \Sigma(W)^V$, we define the *substitution* $t(u) \in \Sigma(W)$ by recursion on t:

$$v \in V \implies v(u) = u_v \quad \text{and} \quad \sigma \in \Sigma,\, t \in \Sigma(V)^{|\sigma|} \implies (\sigma(t))(u) = \sigma(\lambda i.\, t_i(u)) \ . \tag{2.1}$$

Given a theory $\mathbb{T}$ with signature Σ, we define $\mathbb{T}$-*equivalence* as the smallest family of substitution-congruences $\equiv_{\mathbb{T}}$ on the sets $\Sigma(V)$ such that $t \equiv_{\mathbb{T}} u$ for all equations $t = u$ of $\mathbb{T}$. The set $T(V)$ of $\mathbb{T}$-*terms with variables in* V is $\Sigma(V)/{\equiv_{\mathbb{T}}}$.

When a theory $\mathbb{T}$ is seen as specifying a computational effect, $T(V)$ describes the set of computations with effects from $\mathbb{T}$ returning a value in V.

Example 2.3 (Non-deterministic choice)**.** The theory $\mathbb{P}_f^+$ of *non-deterministic choice* comprises a single binary function symbol $\vee$ (written in infix notation) together with the equations

$$x \vee y = y \vee x \qquad\qquad x \vee x = x \qquad\qquad (x \vee y) \vee z = x \vee (y \vee z) \ .$$

The set of terms $P_f^+(V)$ can be identified with the set of non-empty finite subsets of V, where the subset $\{v_1, \dots, v_n\}$ corresponds to the term $v_1 \vee \cdots \vee v_n$. We view this term as encoding a program which chooses non-deterministically between one of the return values $v_1, \dots, v_n$.

Example 2.4 (Probabilistic choice)**.** The theory $\mathbb{P}_f^+$ of *probabilistic choice* comprises a family of binary function symbols $+_r$ indexed by $r \in (0,1)$ together with the equations

$$x +_r y = y +_{r^*} x \qquad\qquad x +_r x = x \qquad\qquad (x +_r y) +_s z = x +_{rs} (y +_{r^* s/(rs)^*} z)$$

where we write r^* for $1 - r$. The set of terms $D(V)$ can be identified with the set of finitely supported discrete probability distributions on V; we see this as a program which chooses probabilistically among possible return values in V.

Example 2.5 (Input)**.** Given a set A, the theory $\mathbb{T}_A$ of A-*valued input* comprises a single A-ary function symbol read, satisfying no equations, whose action we think of as:

$$(t \colon A \to X) \mapsto \mathsf{let\ read()\ be}\ a.\, t(a) \ .$$

The set of terms $T_A(V)$ is, as in the introduction, the initial algebra $\mu X.\, V + X^A$, whose elements may be seen combinatorially as A-ary branching trees with leaves labelled in V; or computationally as programs which request A-values from an external source and use them to determine a return value in V. For example, when $A = \mathbb{N}$, the program which requests two input values and returns their sum is presented by

$$\mathsf{let\ read()\ be}\ n.\, \mathsf{let\ read()\ be}\ m.\, n + m \ \in \ T(\mathbb{N}) \ . \tag{2.2}$$

We now define the models of an algebraic theory. In the definition, we say that a category $\mathcal{C}$ has *powers* if it has all set-indexed self-products $X^A := \Pi_{a \in A} X$.

Definition 2.6 (Σ-structure, $\mathbb{T}$-model)**.** Let Σ be a signature. A Σ-*structure* $\boldsymbol{X}$ in a category $\mathcal{C}$ with powers is an object $X \in \mathcal{C}$ with *operations* $[\![\sigma]\!]_{\boldsymbol{X}} : X^{|\sigma|} \to X$ for each $\sigma \in \Sigma$. For each $t \in \Sigma(V)$ the *derived operation* $[\![t]\!]_{\boldsymbol{X}} : X^V \to X$ is then determined by the recursive clauses:

$$[\![v]\!]_{\boldsymbol{X}} = \pi_v \qquad \text{and} \qquad [\![\sigma(t)]\!]_{\boldsymbol{X}} = X^V \xrightarrow{([\![t_i]\!]_{\boldsymbol{X}})_{i \in |\sigma|}} X^{|\sigma|} \xrightarrow{[\![\sigma]\!]_{\boldsymbol{X}}} X \ . \tag{2.3}$$

Given a theory $\mathbb{T} = (\Sigma, \mathcal{E})$, a $\mathbb{T}$-*model in* $\mathcal{C}$ is a Σ-structure $\boldsymbol{X}$ which satisfies $[\![t]\!]_{\boldsymbol{X}} = [\![u]\!]_{\boldsymbol{X}}$ for all equations $t = u$ of $\mathbb{T}$. The unqualified term "model" will mean "model in *Set*".

A *homomorphism* $f\colon \boldsymbol{X} \to \boldsymbol{Y}$ of $\mathbb{T}$-models in $\mathcal{C}$ is a $\mathcal{C}$-map $f\colon X \to Y$ such that for all $\sigma \in \Sigma$ we have $[\![\sigma]\!]_{\boldsymbol{Y}} \circ f^{|\sigma|} = f \circ [\![\sigma]\!]_{\boldsymbol{X}}$. We write $\mathit{Mod}(\mathbb{T},\mathcal{C})$ for the category of $\mathbb{T}$-models in $\mathcal{C}$, and $\mathit{Mod}(\mathbb{T})$ for the models in Set.

The set of computations $T(V)$ has a structure of $\mathbb{T}$-model $\boldsymbol{T}(V)$ with operations given by substitution; and as is well known, this structure is in fact *free*:

Lemma 2.7. *The inclusion of variables $\eta_V\colon V \to T(V)$ exhibits $\boldsymbol{T}(V)$ as the free $\mathbb{T}$-model on V. That is, for any $\mathbb{T}$-model $\boldsymbol{X}$ and any function $f\colon V \to X$, there is a unique $\mathbb{T}$-model homomorphism $f^\dagger\colon \boldsymbol{T}(V) \to \boldsymbol{X}$ with $f^\dagger \circ \eta_V = f$. Explicitly, $f^\dagger(t) = [\![t]\!]_{\boldsymbol{X}}(\lambda v.\, f(v))$.*

Taking the full subcategory of $\mathit{Mod}(\mathbb{T})$ on the free $\mathbb{T}$-models yields the well known *Kleisli category* of $\mathbb{T}$, which we typically present as follows:

Definition 2.8 (Kleisli category)**.** The *Kleisli category* $\mathit{Kl}(\mathbb{T})$ of a theory $\mathbb{T}$ has sets as objects; hom-sets $\mathit{Kl}(\mathbb{T})(A,B) = \mathit{Set}(A,TB)$; the identity at A being $\eta_A\colon A \to TA$; and composition $g, f \mapsto g^\dagger \circ f$ with $g^\dagger$ as in Lemma 2.7 for the free $\mathbb{T}$-model structures. The *free functor* $F_{\mathbb{T}}\colon \mathit{Set} \to \mathit{Kl}(\mathbb{T})$ is the identity on objects and sends $f \in \mathit{Set}(X,Y)$ to $\eta_Y \circ f \in \mathit{Kl}(\mathbb{T})(X,Y)$. The fully faithful *comparison functor* $I_{\mathbb{T}}\colon \mathit{Kl}(\mathbb{T}) \to \mathit{Mod}(\mathbb{T})$ maps $A \mapsto \boldsymbol{T}A$ and $f \mapsto f^\dagger$.

The Kleisli category captures the compositionality of computations with effects from $\mathbb{T}$, and allows us to draw the link with Moggi's monadic semantics [Mog91]; indeed, the free functor $F_{\mathbb{T}}\colon \mathit{Set} \to \mathit{Kl}(\mathbb{T})$ and its right adjoint $\mathit{Kl}(\mathbb{T})(1,–)\colon \mathit{Kl}(\mathbb{T}) \to \mathit{Set}$ generate an associated monad T on Set and we have that $\mathit{Kl}(\mathbb{T}) \cong \mathit{Kl}(\mathsf{T})$ under Set.

So far we have said nothing about *non*-free $\mathbb{T}$-models. It is a basic fact that every such model can be obtained from a free one by quotienting by some congruence, and so can been seen as a set of computations identified up to some notion of program equivalence. This is important, for example, in [Lev03], and will be important for us in §5 below.

We now turn from models to the dual notion of *comodel*. We say a category $\mathcal{C}$ has *copowers* if if each set-indexed self-coproduct $A \cdot X = \Sigma_{a\in A} X$ exists in $\mathcal{C}$.

Definition 2.9 ($\mathbb{T}$-comodel)**.** Let $\mathbb{T}$ be a theory. A $\mathbb{T}$-*comodel* in a category $\mathcal{C}$ with copowers is a model of $\mathbb{T}$ in $\mathcal{C}^{\mathrm{op}}$, comprising an object $S \in \mathcal{C}$ and *co-operations* $[\![\sigma]\!]^{\boldsymbol{S}}\colon S \to |\sigma|{\cdot}S$ obeying the equations of $\mathbb{T}$. The unqualified term "comodel" will mean "comodel in *Set*". We write $\mathit{Comod}(\mathbb{T},\mathcal{C})$ for the category of $\mathbb{T}$-comodels in $\mathcal{C}$, and $\mathit{Comod}(\mathbb{T})$ for the comodels in Set.

As explained in [PS04, PP08], when a theory $\mathbb{T}$ presents a notion of computation, its comodels provide deterministic environments for evaluating computations with effects from $\mathbb{T}$.

Example 2.10. A comodel $\boldsymbol{S}$ of the theory of A-valued input is a state machine that answers requests for A-characters; it comprises a set of states S and a map $[\![\mathsf{read}]\!]^{\boldsymbol{S}} = (g,n)\colon S \to A{\times}S$ giving for each $s \in S$ a next character $g(s) \in A$ and a next state $n(s) \in S$.

While the comodels of the preceding example are just $A \times (–)$-coalgebras, the comodel perspective adds something to this. The general picture is that a $\mathbb{T}$-comodel allows us to evaluate $\mathbb{T}$-computations $t \in T(V)$ down to values in V via the derived operations of Definition 2.6. Indeed, given a comodel $\boldsymbol{S}$ and a term $t \in T(V)$, we have the derived co-operation $[\![t]\!]^{\boldsymbol{S}}\colon S \to V \times S$ which, unfolding the definition, is given by the clauses:

$$\begin{aligned} v \in V \implies& [\![v]\!]^{\boldsymbol{S}}(s) = (v,s) \\ \text{and } \sigma \in \Sigma, t \in T(V)^{|\sigma|} \implies& [\![\sigma(t)]\!]^{\boldsymbol{S}}(s) = [\![t_i]\!]^{\boldsymbol{S}}(s') \text{ where } [\![\sigma]\!]^{\boldsymbol{S}}(s) = (i,s')\ . \end{aligned} \tag{2.4}$$

The idea is that applying $[\![t]\!]^{\boldsymbol{S}}$ to a starting state $s \in S$ will yield the value $v \in V$ and final state $s' \in S$ obtained by running the computation $t \in T(V)$, using the co-operations of the comodel $\boldsymbol{S}$ to answer the requests posed by the corresponding operation symbols of $\mathbb{T}$.

Example 2.11. For a comodel $(g, n)\colon S \to A \times S$ of A-valued input, the clauses (2.4) become

$$v \in V \implies [\![v]\!]^{\boldsymbol{S}}(s) = (v, s) \qquad t \in T(V)^A \implies [\![\mathsf{read}(t)]\!]^{\boldsymbol{S}}(s) = [\![t(g(s))]\!]^{\boldsymbol{S}}(n(s)) \ .$$

So if we consider $A = \mathbb{N}$, the term $t = \mathsf{read}(\lambda n.\, \mathsf{read}(\lambda m.\, n + m)) \in T(\mathbb{N})$ from (2.2), and the comodel $\boldsymbol{S}$ with $S = \{s, s', s''\}$ and $[\![\mathsf{read}]\!]^{\boldsymbol{S}} = (g, n)\colon S \to \mathbb{N} \times S$ given by the upper line in:

$$\begin{array}{rlll} [\![\mathsf{read}]\!]^{\boldsymbol{S}}: & s \mapsto (3, s') & s' \mapsto (6, s'') & s'' \mapsto (11, s'') \\ [\![t]\!]^{\boldsymbol{S}}: & s \mapsto (9, s'') & s' \mapsto (17, s'') & s'' \mapsto (22, s'') \ , \end{array}$$

then $[\![t]\!]^{\boldsymbol{S}} : S \to \mathbb{N} \times S$ is given by the lower line. For example, we calculate that $[\![t]\!](s) = [\![\mathsf{read}(\lambda n.\, \mathsf{read}(\lambda m.\, n + m))]\!](s) = [\![\mathsf{read}(\lambda m.\, 3 + m)]\!](s') = [\![3 + 6]\!](s'') = (9, s'')$.

As is idiomatic, the *final* comodel of a theory describes "observable behaviours" that states of a comodel may possess. To make this precise, we define states $s_1 \in \boldsymbol{S}_1$ and $s_2 \in \boldsymbol{S}_2$ of two $\mathbb{T}$-comodels to be *operationally equivalent* if running any $\mathbb{T}$-computation $t \in T(V)$ starting from the state s_1 of $\boldsymbol{S}_1$ or from the state s_2 of $\boldsymbol{S}_2$ gives the same value; i.e.,

$$\text{if} \qquad [\![t]\!]^{\boldsymbol{S}_1}(s_1) = (v_1, s_1') \quad \text{and} \quad [\![t]\!]^{\boldsymbol{S}_2}(s_2) = (v_2, s_2') \qquad \text{then} \qquad v_1 = v_2 \ .$$

Lemma 2.12. *States $s_1 \in \boldsymbol{S_1}$ and $s_2 \in \boldsymbol{S}_2$ of two $\mathbb{T}$-comodels are operationally equivalent if and only if they become equal under the unique maps $\boldsymbol{S}_1 \to \boldsymbol{F} \leftarrow \boldsymbol{S}_2$ to the final $\mathbb{T}$-comodel.*

Proof. This is [Gar21, Proposition 5.3]. □

So in the spirit of [KL09, Theorem 4], we may (if we adequately handle the set-theoretic issues) characterise the final $\mathbb{T}$-comodel as the set of all possible states of all possible comodels, modulo operational equivalence. However, a more algebraic approach is also possible. The following is [Gar21, Definition 5.5]:

Definition 2.13 (Admissible behaviour)**.** An *admissible behaviour* β for a theory $\mathbb{T}$ is a family of functions $\beta_V : TV \to V$, as V ranges over sets, such that

$$v \in V \implies \beta_V(v) = v \quad \text{and} \quad t \in TV, u \in (TW)^V \implies \beta_W(t(u)) = \beta_W(t \gg u_{\beta_V(t)}) \ ,$$

where for terms $f \in T(A)$ and $g \in T(B)$, we write $f \gg g$ for the term $f(\lambda a.\, g) \in T(B)$.

An admissible behaviour is a way of evaluating $\mathbb{T}$-computations to values, and from this perspective, the two axioms are quite intuitive: for example, the second says that, if the result of evaluating $t \in TV$ is $v \in V$, then the result of evaluating $t(u) \in TW$ coincides with that of evaluating the computation which sequences t (discarding the return value) into u_v.

Example 2.14. Any state s of a $\mathbb{T}$-comodel $\boldsymbol{S}$ yields an admissible $\mathbb{T}$-behaviour β_s, where for $t \in TV$ we define $\beta_s(t)$ to be the first component of $[\![t]\!]^{\boldsymbol{S}}(s) \in V \times S$.

Proposition 2.15. *The final $\mathbb{T}$-comodel of a theory $\mathbb{T}$ can be described as the set $\boldsymbol{F}$ of admissible behaviours, under the co-operations*

$$[\![\sigma]\!]^{\boldsymbol{F}} : \beta \mapsto (\beta(\sigma), \partial_\sigma \beta) \tag{2.5}$$

where $\partial_\sigma\beta$ is the admissible behaviour given by $(\partial_\sigma\beta)(t) = \beta(\sigma \gg t)$. For any $\mathbb{T}$-comodel $\boldsymbol{S}$, the unique comodel homomorphism $\boldsymbol{S} \to \boldsymbol{F}$ sends s to β_s as in Example 2.14.

Proof. This is [Gar21, Proposition 5.9]. □

For A-valued input, the admissible behaviours can be identified (as in Example 5.10 of *loc. cit.*) with streams of A-values: given an admissible behaviour β, the corresponding stream of values is

$$(\beta(\mathsf{read}), \beta(\mathsf{read} \gg \mathsf{read}), \beta(\mathsf{read} \gg \mathsf{read} \gg \mathsf{read}), \dots) \ .$$

Under this identification, the co-operation $[\![\mathsf{read}]\!]^{\boldsymbol{F}}$ of (2.5) is easily seen to coincide with the structure map (1.1) on the set of A-streams: and in this way, we re-find the familiar construction of the final $\mathbb{T}_A$-comodel as the set of A-streams under (1.1).

The comodel view also allows us to capture the *topology* on the space of streams. Indeed, any comodel of a theory has a natural prodiscrete topology (i.e., topologised as a product of discrete spaces), whose basic open sets describe those states which are indistinguishable with respect to a finite set of $\mathbb{T}$-computations. (This definition appears to be novel.)

Definition 2.16 (Operational topology)**.** Let $\boldsymbol{S}$ be a $\mathbb{T}$-comodel. The *operational topology* on S is generated by sub-basic open sets

$$[t \mapsto v] := \{s \in S : [\![t]\!]^{\boldsymbol{S}}(s) = (v, s') \text{ for some } s'\} \qquad \text{for all } t \in T(V) \text{ and } v \in V \ .$$

Lemma 2.17. *The operational topology makes any $\mathbb{T}$-comodel into a topological comodel. In the case of the final $\mathbb{T}$-comodel, this yields the* final *topological comodel.*

Of course, a topological comodel is simply a comodel in the category *Top* of topological spaces and continuous maps.

Proof. Let $\boldsymbol{S}$ be a $\mathbb{T}$-comodel. Each co-operation $[\![\sigma]\!]^{\boldsymbol{S}} : S \to |\sigma| \cdot S$ is continuous for the operational topology: for indeed, a sub-basic open set of the codomain is $\{i\} \times [t \mapsto v]$, and its inverse image under $[\![\sigma]\!]^{\boldsymbol{S}}$ is the open set

$$\{\, s \in S \mid \exists s', s''.\ [\![\sigma]\!]^{\boldsymbol{S}}(s) = (i, s') \text{ and } [\![t]\!]^{\boldsymbol{S}}(s') = (v, s'')\} = [\sigma \mapsto i] \cap [(\sigma \gg t) \mapsto v] \ , \quad (2.6)$$

where the equality comes from the fact that, if $[\![\sigma]\!]^{\boldsymbol{S}}(s) = (i, s')$, then $[\![t]\!]^{\boldsymbol{S}}(s') = [\![\sigma \gg t]\!]^{\boldsymbol{S}}(s)$.

So the operational topology makes each comodel $\boldsymbol{S}$ into a topological comodel. We now show that, in the case of the final comodel $\boldsymbol{F}$, this yields the final topological comodel. Indeed, if $\boldsymbol{S}$ is *any* topological comodel, we have by finality of $\boldsymbol{F}$ *qua Set*-comodel a unique comodel homomorphism $\beta_{(-)} \colon S \to F$ sending s to the admisssible behaviour β_s, and we need only show this is continuous. But the inverse image under $\beta_{(-)}$ of the subbasic open $[t \mapsto v] \subseteq F$ is the set $\{s \in S : \beta_s(t) = v\} = \{s \in S : [\![t]\!]^{\boldsymbol{S}}(s) = (v, s') \text{ for some } s' \in S\}$, which is open as the inverse image of $\{v\} \times S \subseteq V \times S$ under the continuous map $[\![t]\!]^{\boldsymbol{S}} : S \to V \times S$. □

In the case of the theory of A-valued input, the subbasic open set $[t \mapsto v] \subseteq A^{\mathbb{N}}$ of the final comodel can be defined by induction on $t \in TV$:

$$[w \mapsto v] = \begin{cases} A^{\mathbb{N}} & \text{if } w = v \\ \emptyset & \text{if } w \neq v \in V \end{cases} \qquad \text{and} \qquad [\mathsf{read}(\lambda a.\, t_a) \mapsto v] = \{aW : a \in A, W \in [t_a \mapsto v]\} \ .$$

From this description, we re-find the fact that the final topological comodel is $A^{\mathbb{N}}$ endowed with the product topology for $\mathbb{N}$ copies of the discrete space A.

3. Stream processors as residual comodels

In this section, we recall a more general kind of comodel considered by, among others, [AB20, KRU20, UV20], which allows for stateful translations between different notions of computation. We then explain how this notion allows us to encode stream processors in the sense of [HPG09] and also their *composition*.

Definition 3.1 (Residual comodel)**.** Let $\mathbb{T}$ and $\mathbb{R}$ be theories. An $\mathbb{R}$-*residual* $\mathbb{T}$-*comodel* is a comodel of $\mathbb{T}$ in the Kleisli category $Kl(\mathbb{R})$.

The nomenclature "residual" comes from [KRU20, §5.3], and we will explain the connection to *loc. cit.* in Proposition 3.13 below. For now, let us spell out in detail what a residual comodel $\boldsymbol{S}$ involves. First, there is an underlying set of states S. Next, we have for each operation σ in the signature of $\mathbb{T}$ a basic co-operation $[\![\sigma]\!]^{\boldsymbol{S}} : S \to R(|\sigma| \times S)$ assigning to each state $s \in S$ an $\mathbb{R}$-computation $[\![\sigma]\!]^{\boldsymbol{S}}(s)$ returning values in $|\sigma| \times S$—where, as before, we think of these two components as providing a value answering the request posed by σ, and a next state. Now we recursively determine a derived co-interpretation $[\![t]\!]^{\boldsymbol{S}} : S \to R(V \times S)$ for each $t \in T(V)$ via:

$$\begin{aligned} v \in V \subseteq T(V) &\implies [\![v]\!]^{\boldsymbol{S}}(s) = (v, s) \in V \times S \subseteq R(V \times S) \\ \text{and } \sigma \in \Sigma, t \in T(V)^{|\sigma|} &\implies [\![\sigma(t)]\!]^{\boldsymbol{S}}(s) = [\![\sigma]\!]^{\boldsymbol{S}}(s)\big(\lambda(i, s').\ [\![t_i]\!]^{\boldsymbol{S}}(s')\big) \ , \end{aligned} \tag{3.1}$$

and the final requirement is that these derived operations must satisfy the equations of $\mathbb{T}$.

Remark 3.2. In the second line of (3.1), the element $[\![\sigma]\!]^{\boldsymbol{S}}(s) \in R(|\sigma| \times S)$ is an $\mathbb{R}$-term with variables in $[\![\sigma]\!] \times S$; and substituting each variable $(i, s') \in [\![\sigma]\!] \times S$ therein by the $\mathbb{R}$-term $[\![t_i]\!]^{\boldsymbol{S}}(s') \in R(V)$ gives the value of $[\![\sigma(t)]\!]^{\boldsymbol{S}}(s) \in R(V)$. This amounts to threading $\mathbb{R}$-computations together by *monadic binding*; in Haskell notation, we would write:

$$\begin{aligned} [\![\sigma(t)]\!]^{\boldsymbol{S}}(s) = \textbf{do}\ &(i, s') \leftarrow [\![\sigma]\!]^{\boldsymbol{S}}(s) \\ &\mathsf{return}\ [\![t_i]\!]^{\boldsymbol{S}}(s'). \end{aligned}$$

Example 3.3. A comodel of the theory $\mathbb{T}_B$ of B-valued input residual on the theory $\mathbb{P}_f^+$ of non-deterministic choice comprises a set of states S, and a function $\gamma \colon S \to P_f^+(B \times S)$: thus, a non-terminating, finitely branching labelled transition system.

Example 3.4. A comodel of the theory $\mathbb{T}_B$ of B-valued input residual on the theory $\mathbb{D}$ of probabilistic choice comprises a set of states S, and a function $\gamma \colon S \to D(B \times S)$: thus, a finitely branching probabilistic generative system in the sense of [vGSST90].

However, for us the key example is the following one:

Example 3.5. A comodel of the theory $\mathbb{T}_B$ of B-valued input residual on the theory $\mathbb{T}_A$ of A-valued input comprises a set of states S, and a function $\gamma \colon S \to T_A(B \times S)$ assigning to each state $s \in S$ a program which uses some number of A-tokens from an input stream to inform the choice of an output B-token and a new state in S.

It is easy to see how each state s_0 of such a comodel should encode a stream processor $A^{\mathbb{N}} \to B^{\mathbb{N}}$: given an input stream $\vec{a} \in A^{\mathbb{N}}$, we consume some initial segment $a_0, \dots, a_k$ to answer the requests posed by the program $\gamma(s_0)$, so obtaining an element $b_0 \in B$ and a new state s_1. We now repeat starting from $s_1 \in S$ and the remaining part $\partial^k \vec{a}$ of the input stream, to obtain b_1 and s_2 while consuming $a_{k+1}, \dots, a_\ell$; and so on coinductively. This description was made mathematically precise in [HPG09, §3.1], but in fact we can obtain it in a principled comodel-theoretic manner via (a special case of) a notion given in [PP08, Appendix].

Definition 3.6 (Tensor of a residual comodel with a comodel)**.** Let $\mathbb{T}$ and $\mathbb{R}$ be theories. Let $\boldsymbol{S}$ be an $\mathbb{R}$-residual $\mathbb{T}$-comodel, and let $\boldsymbol{M}$ be an $\mathbb{R}$-comodel. The *tensor product* $\boldsymbol{S} \cdot \boldsymbol{M}$ is the $\mathbb{T}$-comodel with underlying set $S \times M$ and co-operations

$$[\![\sigma]\!]^{\boldsymbol{S}\cdot\boldsymbol{M}} : S \times M \xrightarrow{[\![\sigma]\!]^{\boldsymbol{S}} \times M} R(|\sigma| \times S) \times M \xrightarrow{(t,m)\mapsto [\![t]\!]^{\boldsymbol{M}}(m)} |\sigma| \times S \times M \ . \tag{3.2}$$

This definition makes intuitive sense: given a state machine for translating $\mathbb{T}$-computations into $\mathbb{R}$-computations, and one for executing $\mathbb{R}$-computations, it threads them together to yield a state machine for executing $\mathbb{T}$-computations. We will make this justification rigorous in Definition 3.14 below, but for the moment let us simply assume its reasonability and give:

Definition 3.7 (Trace)**.** Let $\boldsymbol{S}$ be a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel. The *trace* of a state $s \in \boldsymbol{S}$ is the function $\mathsf{tr}(s)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ obtained by partially evaluating at s the unique map of $\mathbb{T}_B$-comodels $\boldsymbol{S} \cdot \boldsymbol{A}^{\mathbb{N}} \to \boldsymbol{B}^{\mathbb{N}}$, where $\boldsymbol{A}^{\mathbb{N}}$ and $\boldsymbol{B}^{\mathbb{N}}$ are endowed with their final comodel structures.

We now unfold this definition. Firstly, for any term $t \in T_A(V)$, the derived co-operation $[\![t]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\vec{a})\colon A^{\mathbb{N}} \to V \times A^{\mathbb{N}}$ is defined recursively by

$$[\![v]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\vec{a}) = (v, \vec{a}) \qquad \text{and} \qquad [\![\mathsf{read}(t)]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\vec{a}) = [\![t_{a_0}]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\partial\vec{a}) \ . \tag{3.3}$$

If we view t as an A-ary branching tree with leaves labelled in V, then $[\![t]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\vec{a})$ is the result of walking up the tree from the root, consuming an element of $\vec{a}$ at each interior node to determine which branch to take, and returning at a leaf the V-value found there along with what remains of $\vec{a}$.

Now, in terms of this, the $\mathbb{T}_B$-comodel structure of $\boldsymbol{S} \cdot \boldsymbol{A}^{\mathbb{N}}$ is given by

$$S \times A^{\mathbb{N}} \to B \times S \times A^{\mathbb{N}} \qquad\qquad (s, \vec{a}) \mapsto [\![\gamma(s)]\!]^{\boldsymbol{A}^{\mathbb{N}}}(\vec{a}) \ ,$$

where $\gamma\colon S \to T_A(B \times S)$ is the residual comodel structure of $\boldsymbol{S}$. This function takes a state s_0 and stream $\vec{a}$ to the triple $(b_0, s_1, \partial^k\vec{a})$ obtained by walking up k nodes of the tree $\gamma(s)$ to the leaf (b_0, s_1). If we view this comodel structure as a triple of maps

$$\mathsf{hd}\colon S \times A^{\mathbb{N}} \to B \qquad \mathsf{next}\colon S \times A^{\mathbb{N}} \to S \qquad \mathsf{tl}\colon S \times A^{\mathbb{N}} \to A^{\mathbb{N}}$$

then we can say, finally, that the trace $\mathsf{tr}(s)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ of $s \in \boldsymbol{S}$ is given coinductively by:

$$\big(\mathsf{tr}(s)(\vec{a})\big)_0 = \mathsf{hd}(s, \vec{a}) \qquad\qquad \partial\big(\mathsf{tr}(s)(\vec{a})\big) = \mathsf{tr}(\mathsf{next}(s, \vec{a}))(\mathsf{tl}(s, \vec{a})) \ .$$

Comparing this construction with that of [HPG09, §3.1], done there with bare hands, we find that they are exactly the same: the derived co-operations $[\![t]\!]$ of (3.3) are the functions *eat t* of *loc. cit.*, while our trace function tr is their function eat_∞.

We have thus shown that each state s of a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel encodes a function $\mathsf{tr}(s)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$; but for these functions to be reasonable stream processors, they should be *continuous* for the profinite topologies. While this may be shown with little effort, we may in fact see it without *any* effort via a comodel-theoretic argument. We first need:

Definition 3.8 (Tensor of a residual comodel and a topological comodel)**.** Let $\mathbb{T}$ and $\mathbb{R}$ be theories, let $\boldsymbol{S}$ be an $\mathbb{R}$-residual $\mathbb{T}$-comodel, and $\boldsymbol{M}$ a topological $\mathbb{R}$-comodel. The *tensor product* $\boldsymbol{S} \cdot \boldsymbol{M}$ is the topological $\mathbb{T}$-comodel with underlying space $S \cdot M$ and co-operations (3.2).

Once again, the justification for this definition will be given below; assuming it for now, the desired continuity of each $\mathsf{tr}(s)$ is immediate. For indeed, viewing $\boldsymbol{A}^{\mathbb{N}}$ and $\boldsymbol{B}^{\mathbb{N}}$ as final topological comodels with the profinite topology, there is a unique map of topological $\mathbb{T}_B$-comodels $\boldsymbol{S} \cdot \boldsymbol{A}^{\mathbb{N}} \to \boldsymbol{B}^{\mathbb{N}}$. Its underlying function is the unique map of *Set*-comodels from Definition 3.7, but the extra information we now gain is the *continuity* of this map—which says precisely that each $\mathsf{tr}(s)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ is continuous, as desired.

The comodel perspective allows us also to say something about *composition* of stream processors. Given a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel $\boldsymbol{S}$, whose states encode the continuous functions $\mathsf{tr}(s)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ for each $s \in \boldsymbol{S}$, and a $\mathbb{T}_B$-residual $\mathbb{T}_C$-comodel $\boldsymbol{P}$, whose states encode the continuous functions $\mathsf{tr}(p)\colon B^{\mathbb{N}} \to C^{\mathbb{N}}$ for each $p \in \boldsymbol{P}$, we may define the *tensor product* $\boldsymbol{P} \cdot \boldsymbol{S}$, which is a $\mathbb{T}_A$-residual $\mathbb{T}_C$-comodel whose states encode precisely the continuous functions $\mathsf{tr}(p) \circ \mathsf{tr}(s)\colon A^{\mathbb{N}} \to C^{\mathbb{N}}$ for $s \in \boldsymbol{S}$ and $p \in \boldsymbol{P}$. The general definition is as follows; again, this will be justified formally by Definition 3.14 below.

Definition 3.9 (Tensor product of two residual comodels)**.** Let $\mathbb{T}$, $\mathbb{R}$ and $\mathbb{V}$ be theories, let $\boldsymbol{S}$ be an $\mathbb{R}$-residual $\mathbb{T}$-comodel and let $\boldsymbol{P}$ be a $\mathbb{T}$-residual $\mathbb{V}$-comodel. The tensor product $\boldsymbol{P} \cdot \boldsymbol{S}$ is the $\mathbb{T}$-residual $\mathbb{V}$-comodel with underlying set $P \times S$ and co-operations

$$[\![\sigma]\!]^{\boldsymbol{P}\cdot\boldsymbol{S}} : P \times S \xrightarrow{[\![\sigma]\!]^{\boldsymbol{P}} \times S} T(|\sigma| \times P) \times S \xrightarrow{(t,s)\mapsto [\![t]\!]^{\boldsymbol{S}}(s)} T(|\sigma| \times P \times S)\ .$$

When in this definition, $\boldsymbol{S}$ is a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel and $\boldsymbol{P}$ is a $\mathbb{T}_B$-residual $\mathbb{T}_C$-comodel, the tensor $\boldsymbol{P} \cdot \boldsymbol{S}$ is the $\mathbb{T}_A$-residual $\mathbb{T}_C$-comodel with underlying set $P \times S$ and structure map

$$[\![\mathsf{read}]\!]^{\boldsymbol{P}\cdot\boldsymbol{S}} : P \times S \xrightarrow{[\![\mathsf{read}]\!]^{\boldsymbol{P}} \times S} T_B(C \times P) \times S \xrightarrow{(t,s)\mapsto [\![t]\!]^{\boldsymbol{S}}(s)} T_A(C \times P \times S)\ . \tag{3.4}$$

To understand this, we must now unfold the definition of $[\![t]\!]^{\boldsymbol{S}}$, which is given by structural recursion over $T_B(C \times P)$ as in (3.1):

$$\begin{aligned} [\![(c,p)]\!]^{\boldsymbol{S}}(s) &= (c,p,s) && \text{for } (c,p) \in C \times P \subseteq T_B(C \times P) \\ [\![\mathsf{read}(\lambda b.\, t_b)]\!]^{\boldsymbol{S}}(s) &= [\![\mathsf{read}]\!]^{\boldsymbol{S}}(s)\big(\lambda(b,s').\ [\![t_b]\!]^{\boldsymbol{S}}(s')\big) && \text{for } t \in T_B(C \times P)^B. \end{aligned} \tag{3.5}$$

Here, in the second clause, $[\![\mathsf{read}]\!]^{\boldsymbol{S}}(s)$ is a term in $T_A(B \times S)$, into which we are substituting the $B \times S$-indexed family of terms $[\![t_b]\!]^{\boldsymbol{S}}(s') \in T_A(C \times P \times S)$ to obtain the desired term in $T_A(C \times P \times S)$.

Let us now see that the states of $\boldsymbol{P} \cdot \boldsymbol{S}$ encode precisely the composites of the continuous functions encoded by the states of $\boldsymbol{P}$ and $\boldsymbol{S}$. Despite the complexity of our description of $\boldsymbol{P} \cdot \boldsymbol{S}$, the proof of this fact is trivial.

Proposition 3.10. *Suppose that $\boldsymbol{S}$ is a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel and $\boldsymbol{P}$ is a $\mathbb{T}_B$-residual $\mathbb{T}_C$-comodel. For any $s \in \boldsymbol{S}$ and $p \in \boldsymbol{P}$, we have that $\mathsf{tr}^{\boldsymbol{P}\cdot\boldsymbol{S}}(p,s) = \mathsf{tr}^{\boldsymbol{P}}(p) \circ \mathsf{tr}^{\boldsymbol{S}}(s)\colon A^{\mathbb{N}} \to C^{\mathbb{N}}$.*

Proof. Consider the diagram of $\mathbb{T}_C$-comodels

$$\begin{array}{ccc} \boldsymbol{P} \cdot \boldsymbol{S} \cdot \boldsymbol{A}^{\mathbb{N}} & \xrightarrow{\boldsymbol{P}\cdot !} & \boldsymbol{P} \cdot \boldsymbol{B}^{\mathbb{N}} \\ & {\scriptstyle !}\searrow \quad \swarrow{\scriptstyle !} & \\ & \boldsymbol{C}^{\mathbb{N}} & \end{array}$$

wherein each map labelled ! is the unique map into a final object. Since $\boldsymbol{C}^{\mathbb{N}}$ is final among $\mathbb{T}_C$-comodels, this diagram clearly commutes; and now partially evaluating at $(p,s) \in P \times S$ yields the desired equality. □

Before continuing, we resolve some unfinished business by justifying Definitions 3.6, 3.8 and 3.9 above. Our starting point will be an alternative presentation of the notion of comodel due to [Uus15]. In *op. cit.*, Uustalu defines a *runner* for a theory $\mathbb{T}$, with set of states S, to be a monad morphism $\mathsf{T} \to \mathsf{T}_S$ from the associated monad of $\mathbb{T}$ to the state monad $\mathsf{T}_S = (- \times S)^S$. The data for such a runner are functions $T(V) \to (V \times S)^S$ assigning to each $t \in T(V)$ a function $[\![t]\!]^{\boldsymbol{S}} : S \to V \times S$. Recognising these as the data of the derived co-operations of a $\mathbb{T}$-comodel structure on S, we should find the main result of [Uus15] reasonable: that $\mathbb{T}$-comodels with underlying set S are in bijection with $\mathbb{T}$-runners with underlying set of states S.

While Uustalu's result is about comodels in *Set*, it generalises unproblematically. For any object S of a category $\mathcal{C}$ with copowers, we have an adjunction $(-) \cdot S \dashv \mathcal{C}(S, -) \colon \mathcal{C} \to Set$ inducing a monad $\mathsf{T}_S = \mathcal{C}(S, (-) \cdot S)$ on *Set*; in [MS14] this is called the *linear-use state monad* associated to S. We now have the following natural extension of Uustalu's result.

Proposition 3.11 [MS14, Theorem 8.2]**.** *Let $\mathbb{T}$ be an algebraic theory, $\mathcal{C}$ a category with copowers, and $S \in \mathcal{C}$. The following are in bijective correspondence:*

(1) *$\mathbb{T}$-comodels $\boldsymbol{S}$ in $\mathcal{C}$ with underlying object S;*
(2) *$\mathbb{T}$-runners in $\mathcal{C}$, i.e., monad maps $[\![-]\!]^{\boldsymbol{S}} : \mathsf{T} \to \mathsf{T}_S$ into the linear-use state monad of S;*
(3) *Functorial extensions of $(-) \cdot S \colon Set \to \mathcal{C}$ along the free functor into the Kleisli category:*

$$\begin{array}{ccc} Set & \xrightarrow{(-)\cdot S} & \mathcal{C} \\ {\scriptstyle F_{\mathbb{T}}}\downarrow & \nearrow_{(-)\cdot \boldsymbol{S}} & \\ Kl(\mathbb{T}) & & \end{array} \qquad (3.6)$$

Remark 3.12. Abstractly, this proposition expresses the fact that $Kl(\mathbb{T})$ is the *free category with copowers containing a comodel of* $\mathbb{T}$; this result is originally due to Linton [Lin66].

Proof. As just said, the argument for (1) $\Leftrightarrow$ (2) is *mutatis mutandis* that of [Uus15, §3]. For (2) $\Leftrightarrow$ (3), it is standard [Mey75] that monad maps $\mathsf{T} \to \mathsf{T}_S$ correspond to extensions to the left in:

$$\begin{array}{ccc} Set & \xrightarrow{F^{\mathsf{T}_S}} & Kl(\mathsf{T}_S) \\ {\scriptstyle F^{\mathsf{T}}}\downarrow & \nearrow & \\ Kl(\mathsf{T}) & & \end{array} \qquad\qquad \begin{array}{ccc} Set & \xrightarrow{(-)\cdot S} & \mathcal{C}_S \\ {\scriptstyle F_{\mathbb{T}}}\downarrow & \nearrow & \\ Kl(\mathbb{T}) & & \end{array}$$

Now the Kleisli category $Kl(\mathsf{T}_S)$ of the linear-use state monad is isomorphic to the category $\mathcal{C}_S$ whose objects are sets, and whose maps $A \to B$ are $\mathcal{C}$-maps $A \cdot S \to B \cdot S$, via an isomorphism which identifies F^{T_S} with $(-) \cdot S \colon Set \to \mathcal{C}_S$. Similarly we have $Kl(\mathsf{T}) \cong Kl(\mathbb{T})$ under *Set*. So monad morphisms $\mathsf{T} \to \mathsf{T}_S$ correspond to extensions as right above: and these, by direct inspection, correspond to extensions as in (3.6). □

If here $\mathcal{C}$ is itself the Kleisli category $Kl(\mathbb{R})$ of a theory $\mathbb{R}$, then the linear-use state monad of $S \in Kl(\mathbb{R})$ is the monad $\mathsf{R}(- \times S)^S$ found as the commuting combination of the state monad for S with the monad R induced by $\mathbb{R}$ (cf. [HPP06, Theorem 10]). Monad maps $\mathsf{T} \to \mathsf{R}(S \times -)^S$ were in [KRU20] termed $\mathbb{R}$-*residual* $\mathbb{T}$-*runners*, and for these the preceding result specialises to:

Proposition 3.13. *Let $\mathbb{T}$ and $\mathbb{R}$ be algebraic theories and let S be a set. The following are in bijective correspondence:*

(1) $\mathbb{R}$-*residual* $\mathbb{T}$-*comodels* $\boldsymbol{S}$ *with underlying set* S*;*
(2) $\mathbb{R}$-*residual* $\mathbb{T}$-*runners* $[\![-]\!]^{\boldsymbol{S}} : \mathsf{T} \to \mathsf{R}(- \times S)^S$*;*
(3) *Functorial extensions of* $(-) \times S \colon \mathit{Set} \to \mathit{Set}$ *through the Kleisli categories of* $\mathbb{T}$ *and* $\mathbb{R}$*:*

$$\begin{array}{ccc} \mathit{Set} & \xrightarrow{(-)\times S} & \mathit{Set} \\ {\scriptstyle F_{\mathbb{T}}}\downarrow & & \downarrow{\scriptstyle F_{\mathbb{R}}} \\ \mathit{Kl}(\mathbb{T}) & \overset{(-)\cdot \boldsymbol{S}}{\dashrightarrow} & \mathit{Kl}(\mathbb{R}) \end{array} . \tag{3.7}$$

By putting together Propositions 3.11 and 3.13, we have an intuitive definition of *tensor product* for a residual comodel and a comodel, or for two residual comodels.

Definition 3.14 (Tensor product of residual comodels)**.** Let $\mathbb{V}$, $\mathbb{T}$, $\mathbb{R}$ be theories; $\boldsymbol{M}$ an $\mathbb{R}$-comodel in $\mathcal{C}$; $\boldsymbol{S}$ an $\mathbb{R}$-residual $\mathbb{T}$-comodel; and $\boldsymbol{P}$ a $\mathbb{T}$-residual $\mathbb{V}$-comodel. The tensor product $\boldsymbol{S} \cdot \boldsymbol{M}$ is the $\mathbb{T}$-comodel in $\mathcal{C}$ classified by the composite of extensions to the left below, while the tensor product $\boldsymbol{P} \cdot \boldsymbol{S}$ is the $\mathbb{R}$-residual $\mathbb{V}$-comodel classified by the composite to the right:

$$\begin{array}{ccccc} \mathit{Set} & \xrightarrow{(-)\times S} & \mathit{Set} & \xrightarrow{(-)\cdot M} & \mathcal{C} \\ {\scriptstyle F_{\mathbb{T}}}\downarrow & & {\scriptstyle F_{\mathbb{R}}}\downarrow & \nearrow{\scriptstyle (-)\cdot \boldsymbol{M}} & \\ \mathit{Kl}(\mathbb{T}) & \overset{(-)\cdot \boldsymbol{S}}{\dashrightarrow} & \mathit{Kl}(\mathbb{R}) & & \end{array} \qquad \begin{array}{ccccc} \mathit{Set} & \xrightarrow{(-)\times P} & \mathit{Set} & \xrightarrow{(-)\times S} & \mathit{Set} \\ {\scriptstyle F_{\mathbb{V}}}\downarrow & & {\scriptstyle F_{\mathbb{T}}}\downarrow & & \downarrow{\scriptstyle F_{\mathbb{R}}} \\ \mathit{Kl}(\mathbb{V}) & \overset{(-)\cdot \boldsymbol{P}}{\dashrightarrow} & \mathit{Kl}(\mathbb{T}) & \overset{(-)\cdot \boldsymbol{S}}{\dashrightarrow} & \mathit{Kl}(\mathbb{R}) \end{array} . \tag{3.8}$$

In particular, when $\mathcal{C} = \mathit{Set}$ and $\mathcal{C} = \mathit{Top}$, the tensor product $\boldsymbol{S} \cdot \boldsymbol{M}$ specialises to those of Definitions 3.6 and 3.8 above; while the tensor product $\boldsymbol{P} \cdot \boldsymbol{S}$ yields Definition 3.9.

Remark 3.15. Here is another perspective on Definition 3.14. To the left of (3.8), the functor $(-) \cdot \boldsymbol{M}$ preserves copowers, and so lifts to a functor $\mathit{Comod}(\mathbb{T}, \mathit{Kl}(\mathbb{R})) \to \mathit{Comod}(\mathbb{T}, \mathcal{C})$, whose value at $\boldsymbol{S}$ is the tensor product $\boldsymbol{S} \cdot \boldsymbol{M}$. We can obtain $\boldsymbol{P} \cdot \boldsymbol{S}$ to the right similarly.

4. Intensional stream processors as a final residual comodel

The arguments of the previous section were given for an arbitrary $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel $\boldsymbol{S}$; but as in [HPG09], it is natural to consider the final residual comodel in particular. To this end, we should first clarify the correct notion of *morphism* between residual comodels.

Definition 4.1 (Map of residual comodels)**.** Let $\mathbb{T}$ and $\mathbb{R}$ be theories, and let $\boldsymbol{S}$ and $\boldsymbol{U}$ be $\mathbb{R}$-residual $\mathbb{T}$-comodels. A map of residual comodels $\boldsymbol{S} \to \boldsymbol{U}$ is a function $f \colon S \to U$ such that $[\![\sigma]\!]^{\boldsymbol{U}} \circ f = R(|\sigma| \times f) \circ [\![\sigma]\!]^{\boldsymbol{S}}$ for all operations σ in the signature of $\mathbb{T}$.

Remark 4.2. Given that an $\mathbb{R}$-residual comodel is a comodel in $\mathit{Kl}(\mathbb{R})$, we might expect a map of residual comodels to be a map in $\mathit{Kl}(\mathbb{R})$, rather than one in Set. The reason for our choice is not pure expediency; it has to do with an enrichment of the category of theories in the category of comonads on Set, currently being investigated by the authors of [KRU20], and which exploits the general *Sweedler theory* of [AJ13]. Working through the calculations, one finds that for two theories $\mathbb{R}$ and $\mathbb{T}$, the category of coalgebras for the hom-comonad $\langle \mathbb{R}, \mathbb{T} \rangle$ is the category of residual comodels, with precisely the maps indicated in Definition 4.1.

With this clarification made, we see that, in particular, the category of $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodels is simply the category of $T_A(B \times -)$-coalgebras, and so we have:

Definition 4.3 (Intensional stream processors)**.** The *type of intensional A-B-stream processors* is the final $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel $\boldsymbol{I_{AB}}$, i.e., the final $T_A(B \times -)$-coalgebra

$$\theta_{AB} \colon I_{AB} \to T_A(B \times I_{AB}) \ . \tag{4.1}$$

The *reflection* function is the trace function of the final residual comodel:

$$\mathsf{reflect} \colon I_{AB} \to \mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}}) \qquad\qquad s \mapsto \mathsf{tr}(s) \colon A^{\mathbb{N}} \to B^{\mathbb{N}} \ ,$$

where here we write *Top* for the category of topological spaces and continuous maps.

As well as reflection, [HPG09] also defines a *reification* function $\mathsf{reify} \colon \mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}}) \to I_{AB}$ that implements each continuous function by a state of the final comodel, and which satisfies $\mathsf{reflect} \circ \mathsf{reify} = \mathrm{id}$. This means that reflect is surjective—but crucially, it is *not* injective. To show this, we must first note that, by the usual techniques, the terminal coalgebra I_{AB} may be described as follows: it is the set of all finite or infinite A-ary branching trees, with interior nodes labelled with elements of B^* (i.e., lists of elements of B), with leaves labelled by elements of $B^{\mathbb{N}}$, and where no infinite path of interior nodes is labelled by the empty list.

Example 4.4. Fix an element $b \in B$ and consider the following two $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel structures on $\{*\}$:

$$\text{(i)}\ * \mapsto (b, *) \qquad\qquad \text{and} \qquad\qquad \text{(ii)}\ * \mapsto \mathsf{read}(\lambda a.\, (b, *)) \ . \tag{4.2}$$

In both comodels, the unique state $*$ encodes the continuous function $A^{\mathbb{N}} \to B^{\mathbb{N}}$ sending every stream $\vec{a}$ to $(b, b, b, b, \dots)$. However, these states yield different elements of the final comodel I_{AB}: (i) gives the trivial tree τ_0 whose root is labelled by $(b, b, b, \dots)$, while (ii) gives the purely infinite A-ary branching tree τ_1 with every node labelled by a single b.

Intuitively, the two states of I_{AB} in this example differ in that the first ignores its input stream entirely, and simply outputs b's without cease; while the second frivolously consumes a single A-token before emitting each b. So I_{AB} is a set of *intensional* representations of stream processors. This will lead us neatly on to the second part of the paper, where we give a comodel-theoretic presentation of *extensional* stream processors, i.e., the set $\mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$, and an explanation in these terms of where the reification function of [HPG09] comes from.

Before we do this, let us see how the tensor product of residual comodels allows us to give an account of the *lazy composition* of intensional stream processors from [HPG09, §4].

Definition 4.5. The *lazy composition* of intensional stream processors is given by the unique map of $\mathbb{T}_A$-residual $\mathbb{T}_C$-comodels $\mathsf{comp} \colon \boldsymbol{I_{BC}} \cdot \boldsymbol{I_{AB}} \to \boldsymbol{I_{AC}}$, where in the domain we take the tensor product of residual comodels of Definition 3.14.

Let us now unpack this definition to see that our composition agrees with the one given in [HPG09, §4]. First, as a special case of (3.4), the $\mathbb{T}_A$-residual $\mathbb{T}_C$-comodel structure on $\boldsymbol{I_{BC}} \cdot \boldsymbol{I_{AB}}$ has co-operation $[\![\mathsf{read}]\!]^{\boldsymbol{I_{BC}} \cdot \boldsymbol{I_{AB}}}$ given by:

$$I_{BC} \times I_{AB} \xrightarrow{\theta_{BC} \times \mathrm{id}} T_B(C \times I_{BC}) \times I_{AB} \xrightarrow{(t,\sigma) \mapsto [\![t]\!]^{\boldsymbol{I_{AB}}}(\sigma)} T_A(C \times I_{BC} \times I_{AB}) \ ,$$

where, as in (4.1), we write θ_{BC} for $[\![\mathsf{read}]\!]^{\boldsymbol{I_{BC}}}$. Furthermore, as in (3.5), we have $[\![t]\!]^{\boldsymbol{I_{AB}}}$ defined recursively by:

$$\begin{aligned} [\![(c, \tau)]\!]^{\boldsymbol{I_{AB}}}(\sigma) &= (c, \tau, \sigma) && \text{for } (c, \tau) \in C \times I_{BC},\ \sigma \in I_{AB} \\ [\![\mathsf{read}(t)]\!]^{\boldsymbol{I_{AB}}}(\sigma) &= \theta_{AB}(\sigma)\big(\lambda(b, \sigma').\ [\![t_b]\!]^{\boldsymbol{I_{AB}}}(\sigma')\big) && \text{for } t \in T_B(C \times I_{BC})^B,\ \sigma \in I_{AB}. \end{aligned} \tag{4.3}$$

We now compare this with the composition function of [HPG09, §4.1], which was obtained as follows. First, the authors define a $\mathbb{T}_A$-residual $\mathbb{T}_C$-comodel structure $\chi\colon S \to T_A(C \times S)$ on the set $S = T_B(C \times I_{BC}) \times T_A(B \times I_{AB})$, via the following clauses:

$$\begin{aligned}
\chi\big((c,\tau),u\big) &= (c, \theta_{BC}(\tau), u) && \text{for } (c,\tau) \in C \times I_{BC}\\
\chi\big(\mathsf{read}(t),(b,\sigma)\big) &= \chi(t_b, \theta_{AB}(\sigma)) && \text{for } t \in T_B(C \times I_{BC})^B,\ (b,\sigma) \in B \times I_{AB}\\
\chi\big(\mathsf{read}(t),\mathsf{read}(u)\big) &= \mathsf{read}(\lambda a.\, \chi(\mathsf{read}(t), u_a)) && \text{for } t \in T_B(C \times I_{BC})^B,\ u \in T_A(B \times I_{AB})^A;
\end{aligned}$$

they now induce by finality of $\boldsymbol{I_{AC}}$ a unique map of residual comodels $u\colon \boldsymbol{S} \to \boldsymbol{I_{AC}}$; and finally, they define the composition map $I_{BC} \times I_{AB} \to I_{AC}$ as $u \circ (\theta_{BC} \times \theta_{AB})$.

Proposition 4.6. *The lazy composition of Definition 4.5 coincides with that of* [HPG09].

Proof. Given the definitions of the two maps, it suffices to show that $\theta_{BC} \times \theta_{AB}$ is a map of residual comodels $\boldsymbol{I_{BC}} \cdot \boldsymbol{I_{AB}} \to \boldsymbol{S}$, i.e., that the outside of

$$\begin{array}{ccccc}
I_{BC} \times I_{AB} & \xrightarrow{\theta_{BC}\times \mathrm{id}} & T_B(C \times I_{BC}) \times I_{AB} & \xrightarrow{(t,\sigma)\mapsto [\![t]\!]^{\boldsymbol{I_{AB}}}(\sigma)} & T_A(C \times I_{BC} \times I_{AB}) \\
{\scriptstyle \theta_{BC}\times\theta_{AB}}\downarrow & \swarrow{\scriptstyle \mathrm{id}\times\theta_{AB}} & & & \downarrow{\scriptstyle T_A(C\times\theta_{BC}\times\theta_{AB})} \\
S & & \xrightarrow{\quad\chi\quad} & & T_A(C \times S)
\end{array}$$

commutes. Clearly the left triangle commutes, so we need only check the same for the right square. From (4.3), the upper composite $f\colon T_B(C \times I_{BC}) \times I_{AB} \to T_A(C \times S)$ around this square is the map defined recursively by

$$\begin{aligned}
f\big((c,\tau),\sigma\big) &= (c, \theta_{BC}(\tau), \theta_{AB}(\sigma)) && \text{for } (c,\tau) \in C \times I_{BC},\ \sigma \in I_{AB}\\
f\big(\mathsf{read}(t),\sigma\big) &= \theta_{AB}(\sigma)\big(\lambda(b,\sigma').\, f(t_b,\sigma')\big) && \text{for } t \in T_B(C \times I_{BC})^B,\ \sigma \in I_{AB}.
\end{aligned}$$

while by (2.1), the clauses defining χ can be rewritten as

$$\begin{aligned}
\chi\big((c,\tau),u\big) &= (c, \theta_{BC}(\tau), u) && \text{for } (c,\tau) \in C \times I_{BC},\ u \in T_A(B \times I_{AB})\\
\chi\big(\mathsf{read}(t),u\big) &= u(\lambda(b,\sigma).\, \chi(t_b, \theta_{AB}(\sigma))) && \text{for } t \in T_B(C \times I_{BC})^B,\ u \in T_A(B \times I_{AB}).
\end{aligned}$$

Comparing these two formulae, it now follows by structural induction on t that $f(t,\sigma) = \chi(t, \theta_{AB}(\sigma))$ for all $(t,\sigma) \in T_B(C \times I_{BC}) \times I_{AB}$, as desired. □

We may now deduce, as in [HPG09, §4.2], that composition of stream processsors corresponds to composition of the underlying continuous functions. Much as in Proposition 3.10, the proof in our setting is trivial.

Proposition 4.7. *The following diagram commutes for all A, B, C:*

$$\begin{array}{ccc}
I_{BC} \times I_{AB} & \xrightarrow{\quad\mathsf{comp}\quad} & I_{AC} \\
{\scriptstyle \mathsf{reflect}\times\mathsf{reflect}}\downarrow & & \downarrow{\scriptstyle \mathsf{reflect}} \\
\mathsf{Top}(B^{\mathbb{N}}, C^{\mathbb{N}}) \times \mathsf{Top}(A^{\mathbb{N}}, B^{\mathbb{N}}) & \xrightarrow{\quad\circ\quad} & \mathsf{Top}(A^{\mathbb{N}}, C^{\mathbb{N}})
\end{array} \qquad (4.4)$$

Proof. Consider the diagram of topological $\mathbb{T}_C$-comodels and comodel homomorphisms:

$$\begin{array}{ccc} \boldsymbol{I_{BC}} \cdot \boldsymbol{I_{AB}} \cdot \boldsymbol{A}^{\mathbb{N}} & \xrightarrow{! \cdot \boldsymbol{A}^{\mathbb{N}}} & \boldsymbol{I_{AC}} \cdot \boldsymbol{A}^{\mathbb{N}} \\ {\scriptstyle \boldsymbol{I_{BC}} \cdot !}\downarrow & & \downarrow{\scriptstyle !} \\ \boldsymbol{I_{BC}} \cdot \boldsymbol{B}^{\mathbb{N}} & \xrightarrow{\quad ! \quad} & \boldsymbol{C}^{\mathbb{N}} \end{array}$$

where each "!" denotes a unique map to a final object. Since $\boldsymbol{C}^{\mathbb{N}}$ is final, this diagram commutes, and currying around the two sides yields the corresponding two sides of (4.4), which thus also commutes. □

Before continuing, let us note that other kinds of residual comodel tensor product are also interesting in this context. For example, if we are given a $\mathbb{P}_f^+$-residual $\mathbb{T}_A$-comodel $\boldsymbol{S}$ (i.e., a labelled transition system with label-set A) together with a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel $\boldsymbol{P}$, then we can tensor them together to get a $\mathbb{P}_f^+$-residual $\mathbb{T}_B$-comodel $\boldsymbol{P} \cdot \boldsymbol{S}$, i.e., a labelled transition system with label-set B; and given a state $s \in \boldsymbol{S}$ and a state $p \in \boldsymbol{P}$, we have the state $(p, s) \in \boldsymbol{P} \cdot \boldsymbol{S}$ in which the transition system $\boldsymbol{S}$ produces a stream of A-labels starting from state s, and feeds them into the stream processor $\boldsymbol{P}$ starting from state p in order to produce a stream of B-values. Just as before, we can calculate explicitly the state machine $\boldsymbol{P} \cdot \boldsymbol{S}$, and see that it is *lazy* in the sense that $\boldsymbol{P}$ requests only the minimal possible number of A-tokens from the transition system $\boldsymbol{S}$ in order to produce each output B-token.

5. Extensional stream processors as a final bimodel

In Section 4, we characterised the set of intensional stream processors as a final $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel. In this section, we give the main result of the paper, characterising the set of extensional stream processors $\mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$ as a final *bimodel* [Fre66, TW70, BH96] for $\mathbb{T}_A$ and $\mathbb{T}_B$.

Definition 5.1 (Bimodel)**.** Let $\mathbb{T}$ and $\mathbb{R}$ be theories. An $\mathbb{R}$-$\mathbb{T}$-*bimodel* $\boldsymbol{K}$ is an $\mathbb{R}$-model $(\boldsymbol{K}, [\![-]\!]_{\boldsymbol{K}})$ endowed with $\mathbb{T}$-comodel structure $[\![-]\!]^{\boldsymbol{K}}$ in the category $\mathit{Mod}(\mathbb{R})$ of $\mathbb{R}$-models.

The main difficulty in working with $\mathbb{R}$-$\mathbb{T}$-bimodels is handling copowers in $\mathit{Mod}(\mathbb{R})$. A simple case is that of *free* $\mathbb{R}$-models: a copower of free models is free, and so we have canonical isomorphisms $B \cdot \boldsymbol{R}(V) \cong \boldsymbol{R}(B \times V)$, which for convenience, we will assume are in fact *identities*, i.e., that the chosen copower $B \cdot \boldsymbol{R}(V)$ is $\boldsymbol{R}(B \times V)$. The $\mathbb{R}$-$\mathbb{T}$-bimodels with free underlying $\mathbb{R}$-model are easy to identify: they correspond precisely to $\mathbb{R}$-residual $\mathbb{T}$-comodels, where the $\mathbb{R}$-$\mathbb{T}$-bimodel $\boldsymbol{R}(\boldsymbol{S})$ corresponding to the $\mathbb{R}$-residual $\mathbb{T}$-comodel $\boldsymbol{S}$ has underlying $\mathbb{R}$-model $\boldsymbol{R}(S)$ and co-operations $[\![\sigma]\!]^{\boldsymbol{R}(\boldsymbol{S})} = \big([\![\sigma]\!]^{\boldsymbol{S}}\big)^{\dagger} \colon \boldsymbol{R}(S) \to \boldsymbol{R}(|\sigma| \times S)$, where $(-)^{\dagger}$ is the Kleisli extension operation of Lemma 2.7.

To understand what we gain by looking at bimodels with non-free underlying model, it is helpful to think in terms of quotients by bisimulations. If $\boldsymbol{S}$ is an $\mathbb{R}$-residual $\mathbb{T}$-comodel, then we could define (cf. [PS15, Definition 5.2]) a *bisimulation* on $\boldsymbol{S}$ to be an equivalence relation $\sim$ on S such that each co-operation $[\![\sigma]\!]^{\boldsymbol{S}}$ sends $\sim$-related states to $\approx$-related computations in $R(|\sigma| \times S)$, where $\approx$ is the congruence generated by $(i, s) \approx (i, s')$ whenever $s \sim s'$. The definition ensures that the residual comodel structure descends to the quotient set $S/{\sim}$; however, this only gives the possibility of identifying operationally equivalent *states*, and not operationally equivalent *computations* over states. The following more generous definition rectifies this.

Definition 5.2 ($\mathbb{R}$-bisimulation)**.** Let $\mathbb{R}$ and $\mathbb{T}$ be theories. For any $\mathbb{R}$-congruence $\sim$ on the free model $\boldsymbol{R}(V)$ and any set B, the congruence $\sim_B$ on $\boldsymbol{R}(B \times V)$ is that generated by

$$t \sim u \text{ in } R(V) \qquad \Longrightarrow \qquad t(\lambda s.(b, s)) \sim_B u(\lambda s.(b, s)) \text{ for all } b \in B.$$

If $\boldsymbol{S}$ is an $\mathbb{R}$-residual $\mathbb{T}$-comodel, then a congruence on $\boldsymbol{R}(S)$ is an $\mathbb{R}$-*bisimulation* if the co-operations $[\![\sigma]\!]^{\boldsymbol{R}(\boldsymbol{S})} = \big([\![\sigma]\!]^{\boldsymbol{S}}\big)^{\dagger} \colon \boldsymbol{R}(S) \to \boldsymbol{R}(|\sigma| \times S)$ of the associated bimodel send $\sim$-congruent terms to $\sim_{|\sigma|}$-congruent terms.

Lemma 5.3. *Let $\boldsymbol{S}$ be an $\mathbb{R}$-residual $\mathbb{T}$-comodel and $\sim$ an $\mathbb{R}$-bisimulation on $\boldsymbol{R}(S)$. There is a unique structure of $\mathbb{R}$-$\mathbb{T}$-bimodel on the quotient $\mathbb{R}$-model $\boldsymbol{K} = \boldsymbol{R}(S)/{\sim}$ for which the the quotient map $q \colon \boldsymbol{R}(S) \twoheadrightarrow \boldsymbol{K}$ becomes a map of bimodels $\boldsymbol{R}(\boldsymbol{S}) \twoheadrightarrow \boldsymbol{K}$.*

Proof. If $\boldsymbol{R}(S)/{\sim} = \boldsymbol{K}$ then $\boldsymbol{R}(B \times S)/{\sim_B}$ is a presentation of the copower $B \cdot \boldsymbol{K}$. So the assumption that $\sim$ is an $\mathbb{R}$-bisimulation ensures that each basic co-operation of $\boldsymbol{R}(\boldsymbol{S})$ descends to a co-operation on $\boldsymbol{K}$, as to the left in:

$$\begin{array}{ccc} \boldsymbol{R}(S) & \xrightarrow{[\![\sigma]\!]^{\boldsymbol{R}(\boldsymbol{S})}} & \boldsymbol{R}(|\sigma| \times S) \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle q_{|\sigma|}} \\ \boldsymbol{K} & \xrightarrow{[\![\sigma]\!]^{\boldsymbol{K}}} & |\sigma| \cdot \boldsymbol{K} \end{array} \qquad\qquad \begin{array}{ccc} \boldsymbol{R}(S) & \xrightarrow{[\![t]\!]^{\boldsymbol{R}(\boldsymbol{S})}} & \boldsymbol{R}(V \times S) \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle q_V} \\ \boldsymbol{K} & \xrightarrow{[\![t]\!]^{\boldsymbol{K}}} & V \cdot \boldsymbol{K} \end{array} .$$

Since $\boldsymbol{R}(|\sigma| \times S)$ is the copower $|\sigma| \cdot \boldsymbol{R}(S)$, and the quotient map $q_{|\sigma|}$ is the copower $|\sigma| \cdot q$, it follows that the *derived* co-operations of $\boldsymbol{R}(\boldsymbol{S})$ descend to the corresponding *derived* co-operations on $\boldsymbol{K}$, as to the right above; whence the satisfaction of the $\mathbb{T}$-comodel equations for $\boldsymbol{R}(\boldsymbol{S})$ implies the corresponding satisfaction for $\boldsymbol{K}$. So $\boldsymbol{K}$ is an $\mathbb{R}$-$\mathbb{T}$-bimodel, and clearly this is the *unique* bimodel structure making q into a bimodel homomorphism. □

We can use this construction to explain how the passage from the final $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel to the final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel will collapse the intensionality we saw in Example 4.4.

Example 5.4. Consider the two $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodels $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ of Example 4.4. While there is clearly no scope for quotienting by a bisimulation on the set of states $\{*\}$, we *can* non-trivially quotient each by a $\mathbb{T}_A$-bisimulation on $\boldsymbol{T}_A(*)$: namely the $\mathbb{T}_A$-congruence on $\boldsymbol{T}_A(*)$ generated by $* \sim \mathsf{read}(\lambda a.\, *)$. It is easy to see that this is a $\mathbb{T}_A$-bisimulation for both $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$, and so we obtain quotient $\mathbb{T}_A$-$\mathbb{T}_B$-bimodels $\boldsymbol{T}_A(\boldsymbol{S}_1)/{\sim}$ and $\boldsymbol{T}_A(\boldsymbol{S}_2)/{\sim}$. In fact, these are visibly the *same* bimodel $\boldsymbol{K}$, with underlying $\mathbb{T}_A$-model the final model $\{*\}$, and with $\mathbb{T}_B$-comodel structure $[\![\mathsf{read}]\!]^{\boldsymbol{K}} : \boldsymbol{K} \to B \cdot \boldsymbol{K}$ given by the bth coproduct injection. So we have a cospan of bimodels $\boldsymbol{T}_A(\boldsymbol{S}_1) \twoheadrightarrow \boldsymbol{K} \twoheadleftarrow \boldsymbol{T}_A(\boldsymbol{S}_2)$, which in particular implies that the states $*$ of $\boldsymbol{T}_A(\boldsymbol{S}_1)$ and $\boldsymbol{T}_A(\boldsymbol{S}_2)$ must be identified in a *final* $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel.

This example provides supporting evidence for the main theorem we shall now prove: that the set $\mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$ of extensional stream processors is a final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel. Before giving this, let us mention some related results:

Proposition 5.5. *For any set B, the final $\mathbb{P}_f^+$-$\mathbb{T}_B$-bimodel is given by the set of topologically closed subsets of $B^{\mathbb{N}}$, with the $\mathbb{P}_f^+$-model structure given by binary union, and with $\mathbb{T}_B$-comodel structure given as in* [Gar18, Proposition 71].

Proposition 5.6. *For any set B, the final $\mathbb{D}$-$\mathbb{T}_B$-bimodel is given by the set of probability distributions on $B^{\mathbb{N}}$, with the $\mathbb{D}$-model structure given by the usual convex combination of probability distributions, and with $\mathbb{T}_B$-comodel structure given as in* [Gar18, Proposition 79].

In both of these examples, the elements of the final bimodel can be seen as *traces* for states of the corresponding residual comodels (which, we recall, are non-terminating labelled transition systems and finitely branching probabilistic generative systems respectively.) It is thus consistent for us to think of the final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel as providing an "object of traces" for A-B-stream processors.

To show that this final bimodel can be identified with $\mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$, we will first construct an adjunction as to the left in

$$\mathit{Mod}(\mathbb{T}_A) \underset{(-)\otimes \boldsymbol{A}^{\mathbb{N}}}{\overset{\mathit{Top}(\boldsymbol{A}^{\mathbb{N}},-)}{\underset{\longrightarrow}{\overset{\longleftarrow}{\top}}}} \mathit{Top} \qquad \mathit{Comod}(\mathbb{T}_B, \mathit{Mod}(\mathbb{T}_A)) \underset{(-)\otimes \boldsymbol{A}^{\mathbb{N}}}{\overset{\mathit{Top}(\boldsymbol{A}^{\mathbb{N}},-)}{\underset{\longrightarrow}{\overset{\longleftarrow}{\top}}}} \mathit{Comod}(\mathbb{T}_B, \mathit{Top}) \tag{5.1}$$

We then show that *both* directions of this adjunction preserve coproducts, so in particular copowers; it will then follow that the adjunction to the left lifts to one as to the right on $\mathbb{T}_B$-comodels. The right adjoint of this lifted adjunction, like any right adjoint, will preserve terminal objects, and so must send the final topological $\mathbb{T}_B$-comodel $\boldsymbol{B}^{\mathbb{N}}$ to a final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel, with underlying set $\mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$.

To construct the adjunction to the left in (5.1) we apply a standard result of category-theoretic universal algebra (cf. [Fre66, Theorem 2]). For self-containedness we give a full proof.

Proposition 5.7. *Let $\mathcal{C}$ be a category with copowers and $\boldsymbol{S}$ a $\mathbb{T}$-comodel in $\mathcal{C}$. For any object $C \in \mathcal{C}$, the hom-set $\mathcal{C}(S, C)$ bears a structure of $\mathbb{T}$-model $\mathcal{C}(\boldsymbol{S}, C)$ with operations*

$$[\![\sigma]\!]_{\mathcal{C}(\boldsymbol{S},C)}\,(\lambda i.\, S \xrightarrow{f_i} C) = S \xrightarrow{[\![\sigma]\!]^{\boldsymbol{S}}} |\sigma| \cdot S \xrightarrow{\langle f_i \rangle_{i\in|\sigma|}} C \tag{5.2}$$

where $\langle f_i\rangle_{i\in|\sigma|}$ is the copairing of the f_i's. As C varies, this assignment underlies a functor $\mathcal{C}(\boldsymbol{S}, –)\colon \mathcal{C} \to \mathit{Mod}(\mathbb{T})$. If $\mathcal{C}$ is cocomplete, this functor has a left adjoint $(–)\otimes\boldsymbol{S}\colon \mathit{Mod}(\mathbb{T}) \to \mathcal{C}$.

Proof. For any $C \in \mathcal{C}$, the hom-functor $\mathcal{C}(–, C)\colon \mathcal{C}^{\mathrm{op}} \to \mathit{Set}$ sends copowers in $\mathcal{C}$ to powers in Set, and so sends $\mathbb{T}$-comodels in $\mathcal{C}$ to $\mathbb{T}$-models in Set. In particular, the $\mathbb{T}$-model induced by $\boldsymbol{S} \in \mathit{Comod}(\mathbb{T}, \mathcal{C})$ is $\mathcal{C}(\boldsymbol{S}, C)$ with the operations defined above. The functoriality of this assignment is clear, so it remains to exhibit the desired adjoint when $\mathcal{C}$ is cocomplete.

To this end, note that a $\mathbb{T}$-model homomorphism $\alpha\colon \boldsymbol{X} \to \mathcal{C}(\boldsymbol{S}, C)$ is equally a function $\alpha\colon X \to \mathcal{C}(S, C)$ such that, for all basic $\mathbb{T}$-operations σ and all $\vec{x} \in X^{|\sigma|}$, we have

$$S \xrightarrow{\alpha([\![\sigma]\!]_{\boldsymbol{X}}(\vec{x}))} C \qquad = \qquad S \xrightarrow{[\![\sigma]\!]^{\boldsymbol{S}}} |\sigma| \cdot S \xrightarrow{\langle \alpha(x_i)\rangle_{i\in|\sigma|}} C\ .$$

Transposing under $(–) \cdot S \dashv \mathcal{C}(S, –)\colon \mathcal{C} \to \mathit{Set}$, this is equally to give a map $\bar{\alpha}\colon X \cdot S \to C$ in $\mathcal{C}$ such that, for each basic $\mathbb{T}$-operation σ, postcomposition with $\bar{\alpha}$ equalises the two maps

$$X^{|\sigma|} \cdot S \xrightarrow{[\![\sigma]\!]_{\boldsymbol{X}} \cdot C} X \cdot S \qquad X^{|\sigma|} \cdot S \xrightarrow{X^{|\sigma|} \cdot [\![\sigma]\!]^{\boldsymbol{X}}} X^{|\sigma|} \cdot (|\sigma| \cdot S) \cong (X^{|\sigma|} \times |\sigma|) \cdot S \xrightarrow{\mathrm{ev}\cdot S} X \cdot S\ .$$

Thus, defining $\boldsymbol{X} \otimes \boldsymbol{S}$ to be the joint coequaliser of these parallel pairs as σ varies across the basic $\mathbb{T}$-operations, we have bijections $\mathcal{C}(\boldsymbol{X} \otimes \boldsymbol{S}, C) \cong \mathit{Mod}(\mathbb{T})(\boldsymbol{X}, \mathcal{C}(\boldsymbol{S}, C))$ natural in $C \in \mathcal{C}$, so that $\boldsymbol{X} \otimes \boldsymbol{S}$ is the value at $\boldsymbol{X}$ of the desired left adjoint $(–) \otimes \boldsymbol{S}$. □

Remark 5.8. Again, the final part of this result expresses an abstract fact: $\mathit{Mod}(\mathbb{T})$ is the *free cocomplete category containing a comodel of* $\mathbb{T}$.

In particular, we may apply the preceding result when $\mathcal{C}$ is the cocomplete category *Top* and $\boldsymbol{S}$ is the final topological $\mathbb{T}_A$-comodel $\boldsymbol{A}^{\mathbb{N}}$ to obtain an adjunction as to the left in (5.1). We now show that both directions of this adjunction preserve coproducts, and so in particular copowers. Since left adjoints always preserve colimits, there is only work to do for the right adjoint $\mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, –)\colon \mathit{Top} \to \mathit{Mod}(\mathbb{T}_A)$. First we spell out that, on objects, this functor acts by taking a space C to the set of continuous functions $\mathit{Top}(A^{\mathbb{N}}, C)$, under the A-ary magma structure split that takes a family $(f_a : a \in A)$ of functions to the function $\mathsf{split}(\vec{f})$ with

$$\mathsf{split}(\vec{f})(\vec{a}) = f_{a_0}(\partial\vec{a}) \ . \tag{5.3}$$

In other words, $\mathsf{split}(\vec{f})$ consumes the first token a_0 of its input and then continues as f_{a_0} on the rest of its input; note that split is in fact invertible, with inverse given by the function $\mathsf{split}^{-1}(f) = (f(a–) : a \in A)$. This describes the action of $\mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, –)\colon \mathit{Top} \to \mathit{Mod}(\mathbb{T}_A)$ on objects; on morphisms, it simply acts by postcomposition.

The following result is the main piece of serious work needed to complete our result; it refines the topological arguments described in [HPG09, Theorem 2.1], and used there to construct the reification function for intensional stream processors.

Proposition 5.9. *The functor $\mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, –)\colon \mathit{Top} \to \mathit{Mod}(\mathbb{T}_A)$ preserves coproducts.*

Proof. Given spaces $(X_i : i \in I)$, we have the coproduct injections $\iota_i\colon X_i \to \Sigma_i X_i$ in *Top*, and must show that the family of postcomposition maps

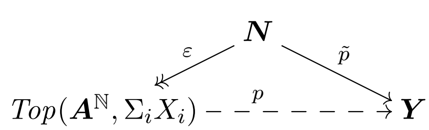
$$\big(\, \iota_i \circ (–)\colon \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, X_i) \to \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, \Sigma_i X_i)\,\big)_{i \in I} \tag{5.4}$$

constitute a coproduct cocone in $\mathit{Mod}(\mathbb{T}_A)$. We first show:

Lemma 5.10. *The maps* (5.4) *are jointly epimorphic in $\mathit{Mod}(\mathbb{T}_A)$.*

Proof. We show that the sub-A-ary magma $\boldsymbol{M} \subseteq \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, \Sigma_i X_i)$ generated by the image of the maps (5.4) is all of $\mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, \Sigma_i X_i)$. So suppose not; then there exists some continuous $f\colon A^{\mathbb{N}} \to \Sigma_i X_i$ with $f \notin M$. Since we have $f = \mathsf{split}(\mathsf{split}^{-1}(f)) = \mathsf{split}(\lambda a.\, f(a–))$, we can find $a_0 \in A$ with $f(a_0–) \notin M$. Now repeating the argument with $f(a_0–)$, we can find $a_1 \in A$ with $f(a_0a_1–) \notin M$; and continuing in this fashion, making countably many dependent choices, we find some $\vec{a} \in A^{\mathbb{N}}$ such that for all n, the continuous function $f(a_0a_1\dots a_n–)\colon A^{\mathbb{N}} \to \Sigma_i X_i$ is not in M. In particular, none of these functions factor through any X_i; but as $f(\vec{a}) \in X_i$ for some i, this means there is *no* open neighbourhood of $\vec{a}$ which is mapped by f into the open neigbourhood X_i of $f(\vec{a})$, contradicting the continuity of f. □

Thus, to complete the proof, we need only show that, for a cocone $(p_i\colon \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, X_i) \to \boldsymbol{Y})_{i \in I}$ in $\mathit{Mod}(\mathbb{T}_A)$, there exists *some* map $p\colon \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, \Sigma_i X_i) \to \boldsymbol{Y}$ with $p \circ \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, \iota_i) = p_i$ for each i. To this end, consider the diagram of A-ary magmas



where $\boldsymbol{N} = (N, \nu)$ is the free A-ary magma generated by symbols $[f, i]$ for $i \in I$ and $f \in \mathit{Top}(\boldsymbol{A}^{\mathbb{N}}, X_i)$, where ε sends $[f, i]$ to $\iota_i f$ and where $\tilde{p}$ sends $[f, i]$ to $p_i(f)$. It suffices to exhibit a factorisation p of $\tilde{p}$ through ε as displayed. Now by the lemma above, ε is epimorphic, and so the coequaliser of its kernel-congruence; so to obtain such a factorisation, it suffices to show that if $x, y \in N$ satisfy $\varepsilon(x) = \varepsilon(y)$, then they satisfy $\tilde{p}(x) = \tilde{p}(y)$. We do so by induction on the total number of magma operations ν in x and y:

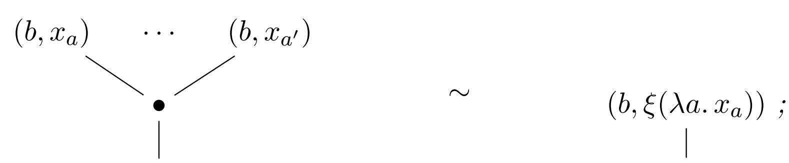
- If $x = [f, i]$ and $y = [g, j]$ then $\varepsilon(x) = \varepsilon(y)$ says that $\iota_i f = \iota_j g$, which is possible only if $i = j$ and $f = g$. So $x = y$ and so certainly $\tilde{p}(x) = \tilde{p}(y)$.
- If $x = [f, i]$ and $y = \nu(\lambda a.\, y_a)$ then on taking $x_a = [f(a\text{–}), i]$ for each a, we get from $\varepsilon(x) = \varepsilon(y)$ that
$$\mathsf{split}(\lambda a.\, \varepsilon(x_a)) = \mathsf{split}(\lambda a.\, \iota_i f(a\text{–})) = \iota_i f = \varepsilon(x) = \varepsilon(y) = \varepsilon(\nu(\lambda a.\, y_a)) = \mathsf{split}(\lambda a.\, \varepsilon(y_a))$$
which, since split is invertible, implies that $\varepsilon(x_a) = \varepsilon(y_a)$ for each $a \in A$. By induction, we have $\tilde{p}(x_a) = \tilde{p}(y_a)$ for each a, and so we have the desired equality:
$$\tilde{p}(x) = p_i(f) = \mathsf{split}(\lambda a.\, p_i(f(a\text{–}))) = \mathsf{split}(\lambda a.\, \tilde{p}(x_a)) = \mathsf{split}(\lambda a.\, \tilde{p}(y_a)) = \tilde{p}(\nu(\lambda a.\, y_a) = \tilde{p}(y)\ .$$
- The case where $x = \nu(\lambda a.\, x_a)$ and $y = [g, j]$ is dual.
- Finally, if $x = \nu(\lambda a.\, x_a)$ and $y = \nu(\lambda a.\, y_a)$, then from $\varepsilon(x) = \varepsilon(y)$ we get
$$\mathsf{split}(\lambda a.\, \varepsilon(x_a)) = \varepsilon(\nu(\lambda a.\, x_a)) = \varepsilon(x) = \varepsilon(y) = \varepsilon(\nu(\lambda a.\, y_a)) = \mathsf{split}(\lambda a.\, \varepsilon(y_a))$$
and so by invertibility of split that $\varepsilon(x_a) = \varepsilon(y_a)$ for all a. By induction, $\tilde{p}(x_a) = \tilde{p}(y_a)$ for all a, and so the desired equality
$$\tilde{p}(x) = \tilde{p}(\nu(\lambda a.\, x_a) = \mathsf{split}(\lambda a.\, \tilde{p}(x_a)) = \mathsf{split}(\lambda a.\, \tilde{p}(y_a)) = \tilde{p}(\nu(\lambda a.\, y_a) = \tilde{p}(y)\ . \qquad \square$$

Using this result, we can conclude the argument as explained above. Since both adjoints to the left of (5.1) preserve coproducts, the adjunction lifts to an adjunction between categories of $\mathbb{T}_B$-comodels as to the right. In particular, the lifted right adjoint sends the final topological $\mathbb{T}_B$-comodel to a final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel, so giving our main theorem:

Theorem 5.11. *For any sets A and B, the final $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel $\boldsymbol{E_{AB}}$ is given by the set of continuous functions $Top(A^{\mathbb{N}}, B^{\mathbb{N}})$ with the $\mathbb{T}_A$-model structure of* (5.3)*, and with the $\mathbb{T}_B$-comodel structure map*
$$Top(\boldsymbol{A}^{\mathbb{N}}, B^{\mathbb{N}}) \xrightarrow{(g,n)\circ(-)} Top(\boldsymbol{A}^{\mathbb{N}}, B \cdot B^{\mathbb{N}}) \xrightarrow{\cong} B \cdot Top(\boldsymbol{A}^{\mathbb{N}}, B^{\mathbb{N}})\ , \tag{5.5}$$
whose first part is postcomposition with (1.1) *and whose second part is the canonical isomorphism coming from the fact that $Top(\boldsymbol{A}^{\mathbb{N}}, \text{–})\colon Top \to Mod(\mathbb{T}_A)$ preserves coproducts.*

We now describe (5.5) more concretely, but first we describe copowers in $Mod(\mathbb{T}_A)$.

Lemma 5.12. *For any $\mathbb{T}_A$-model $\boldsymbol{X} = (X, \xi)$ and set B, the copower $B \cdot \boldsymbol{X}$ may be found as either: (i) the quotient of $\boldsymbol{T}_A(B \times X)$ by the congruence which identifies*



(5.6)

or: (ii) the subset of $T_A(B \times X)$ on those A-ary branching trees where no non-trivial subtree has all its leaves labelled by the same element of B, with the $\mathbb{T}_A$-model structure map υ being that of $T_A(B \times X)$ except that $\upsilon(\lambda a.\, (b, x_a)) = (b, \xi(\lambda a.\, x_a))$.

Proof. (i) is the presentation $\boldsymbol{T}_A(B \times X)/{\sim_B}$ from Lemma 5.3 when $\sim$ is the congruence associated to the quotient $\mathrm{id}^{\dagger}\colon \boldsymbol{T}_A(X) \twoheadrightarrow \boldsymbol{X}$. As for (ii), these elements are the normal forms for the strongly normalising rewrite system obtained by applying (5.6) from left to right. $\square$

Via presentation (i), we may thus describe (5.5) by associating to each $f \in Top(A^{\mathbb{N}}, B^{\mathbb{N}})$ a suitable tree in $T_A(B \times Top(A^{\mathbb{N}}, B^{\mathbb{N}}))$. For this, we use the identification of $B^{\mathbb{N}}$ with $B \cdot B^{\mathbb{N}}$ via $\vec{b} \mapsto (b_0, \partial\vec{b})$, together with Lemma 5.10, to see that f lies in the closure under the A-ary magma operation split on $Top(A^{\mathbb{N}}, B^{\mathbb{N}})$ of the set of those $g\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ for which $g(\vec{a})_0$ is constant. (This expresses algebraically the fact that, for each $\vec{a} \in A^{\mathbb{N}}$, there is some finite initial segment $a_0 \dots a_k$ of $\vec{a}$ such that $f(\vec{a}')_0 = f(\vec{a})_0$ whenever $a_0 \dots a_k = a'_0 \dots a'_k$.)

Choosing any such presentation of f gives a well-founded A-ary tree (encoding the applications of split) with leaves labelled by functions $g\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$ with $g(\vec{a})_0$ constant. Each such g is equally specified by the constant $b = g(\vec{a})_0$, and the function $h = \partial \circ g\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$, so that our leaf labels are equally elements in $B \times Top(A^{\mathbb{N}}, B^{\mathbb{N}})$: so altogether we have an element of $T_A(B \times Top(A^{\mathbb{N}}, B^{\mathbb{N}}))$. Note that choosing a different presentation of f would yield a different element of $T_A(B \times Top(A^{\mathbb{N}}, B^{\mathbb{N}}))$; however, our theory ensures that these elements are congruent under (5.6), so yielding a well-defined element of $B \cdot Top(\boldsymbol{A}^{\mathbb{N}}, B^{\mathbb{N}})$.

6. Comparing intensional and extensional stream processors

To conclude the paper, we examine the unique maps from an arbitrary $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel to the final one, showing that these act as expected via the trace function of Definition 3.7; and, finally, we give a comodel-theoretic explanation of "normalisation-by-trace-evaluation" for intensional stream processors.

We begin with a small refinement of Proposition 5.7.

Proposition 6.1 [PP08, Theorem 4.4]**.** *Let $\mathcal{C}$ be a cocomplete category and $\boldsymbol{S}$ a $\mathbb{T}$-comodel in $\mathcal{C}$. The functor $(\text{–}) \otimes \boldsymbol{S}$ of Proposition 5.7 may be chosen to render commutative the following diagram, whose top edge is as in* (3.6)*, and whose left edge is as in Definition 2.8:*

$$\begin{array}{ccc} Kl(\mathbb{T}) & \xrightarrow{(\text{–})\cdot \boldsymbol{S}} & \mathcal{C} \\ {\scriptstyle I_{\mathbb{T}}}\big\downarrow & \nearrow_{(\text{–})\otimes \boldsymbol{S}} & \\ Mod(\mathbb{T}) & & \end{array} \quad . \tag{6.1}$$

Proof. For a free $\mathbb{T}$-model $\boldsymbol{T}(V)$, we have natural bijections $Mod(\mathbb{T})(\boldsymbol{T}(V), \mathcal{C}(\boldsymbol{S}, C)) \cong Set(V, \mathcal{C}(S, C)) \cong \mathcal{C}(V \cdot S, C)$, and so we may take $\boldsymbol{T}(V) \otimes \boldsymbol{S} = V \cdot S$. This makes (6.1) commute on objects. On morphisms, given $\theta^\dagger\colon \boldsymbol{T}(V) \to \boldsymbol{T}(W)$ in $Mod(\mathbb{T})$, its image $\theta^\dagger \otimes \boldsymbol{S}\colon V \cdot S \to W \cdot S$ under $(\text{–}) \otimes \boldsymbol{S}$ is, by adjointness, the unique map making:

$$\begin{array}{ccccc} \mathcal{C}(W \cdot S, C) & \xrightarrow{\cong} & Set(W, \mathcal{C}(S, C)) & \xrightarrow{\cong} & Mod(\mathbb{T})(\boldsymbol{T}W, \mathcal{C}(\boldsymbol{S}, C)) \\ {\scriptstyle (\text{–})\circ(\theta^\dagger \otimes \boldsymbol{S})}\big\downarrow & & \vdots & & \big\downarrow{\scriptstyle (\text{–})\circ\theta^+} \\ \mathcal{C}(V \cdot S, C) & \xrightarrow{\cong} & Set(V, \mathcal{C}(S, C)) & \xrightarrow{\cong} & Mod(\mathbb{T})(\boldsymbol{T}V, \mathcal{C}(\boldsymbol{S}, C)) \end{array}$$

commute for all $C \in \mathcal{C}$. Now, the unique dotted map making the right square commute is, by the freeness of $\boldsymbol{T}(V)$, the function

$$(f_w \in \mathcal{C}(S, C) : w \in W) \qquad \mapsto \qquad (\llbracket \theta(v) \rrbracket_{\mathcal{C}(\boldsymbol{S},C)}(\vec{f}) : v \in V) \ .$$

But by induction on (5.2), we have $\llbracket \theta(v) \rrbracket_{\mathcal{C}(\boldsymbol{S},C)}(\vec{f}) = S \xrightarrow{\llbracket \theta(v) \rrbracket^{\boldsymbol{S}}} W \cdot S \xrightarrow{\langle f_i \rangle_{i \in |\sigma|}} C$; whence we must have $\theta^\dagger \otimes \boldsymbol{S} = \langle \llbracket \theta(v) \rrbracket^{\boldsymbol{S}} \rangle_{v \in V} = \theta \cdot \boldsymbol{S}$ as desired. ∎

We now characterise the unique maps to $\boldsymbol{E_{AB}}$ from bimodels induced by residual comodels.

Proposition 6.2. *Let $\boldsymbol{S}$ be a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel. The unique bimodel map from the associated bimodel $f\colon \boldsymbol{T}_A(\boldsymbol{S}) \to \boldsymbol{E_{AB}}$ is $\mathsf{tr}^\dagger$, the homomorphic extension of the trace function $\mathsf{tr}\colon S \to \mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$ of Definition 3.7.*

Proof. Since $(\text{–}) \otimes \boldsymbol{A}^{\mathbb{N}}\colon \mathit{Mod}(\mathbb{T}_A) \to \mathit{Top}$ restricts back along $I_{\mathbb{T}_A}$ to $(\text{–}) \cdot \boldsymbol{A}^{\mathbb{N}}\colon \mathit{Kl}(\mathbb{T}_A) \to \mathit{Top}$, its lifting to a functor on $\mathbb{T}_B$-comodels must, by Remark 3.15, restrict along $I_{\mathbb{T}_A}$ to the tensor product of Definition 3.14. So the unique $\mathbb{T}_B$-comodel map $\boldsymbol{T}_A(\boldsymbol{S}) \otimes \boldsymbol{A}^{\mathbb{N}} \to \boldsymbol{B}^{\mathbb{N}}$ must be the unique map $\boldsymbol{S} \cdot \boldsymbol{A}^{\mathbb{N}} \to \boldsymbol{B}^{\mathbb{N}}$ of Definition 3.7. By the proof of Proposition 6.1, transposing this latter map to a bimodel map $\boldsymbol{T}_A(\boldsymbol{S}) \to \boldsymbol{E_{AB}}$ is achieved by first currying—which yields the trace function $\mathsf{tr}\colon S \to \mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$—and then extending homomorphically. □

We now do the same for the unique maps to $\boldsymbol{E_{AB}}$ from *arbitrary* $\mathbb{T}_A$-$\mathbb{T}_B$-bimodels. To do so, we show that every such bimodel arises in a canonical way from the construction of Lemma 5.3. Note that this is *not* true for bimodels over arbitrary theories; it relies on a special property of the theory $\mathbb{T}_A$, namely that it admits *abstract hypernormalisation* in the sense of [Gar18]. See section 7 of *op. cit.* for a more detailed explanation of this phenomenon.

Lemma 6.3. *Let $\boldsymbol{K}$ be a $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel. The composite*

$$\gamma = K \xrightarrow{[\![\mathsf{read}]\!]^{\boldsymbol{K}}} B \cdot K \xrightarrow{\subseteq} T_A(B \times K)$$

where we take $B \cdot \boldsymbol{K} \subseteq T_A(B \times K)$ as in Lemma 5.12(ii), endows K with the structure of a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel $\check{\boldsymbol{K}}$. The congruence on $\boldsymbol{T}_A(K)$ generating the $\mathbb{T}_A$-model quotient map $\mathrm{id}_K^\dagger\colon \boldsymbol{T}_A(K) \twoheadrightarrow \boldsymbol{K}$ is a $\mathbb{T}_A$-bisimulation for $\check{\boldsymbol{K}}$ and the quotient bimodel is precisely $\boldsymbol{K}$.

Proof. Only the final sentence requires any verification; it will follow if we can show that the square of $\mathbb{T}_A$-model maps to the left below is commutative:

$$\begin{array}{ccc} \boldsymbol{T}_A(K) & \xrightarrow{\gamma^\dagger} & \boldsymbol{T}_A(B \times K) \\ {\scriptstyle \mathrm{id}_K^\dagger}\downarrow & & \downarrow{\scriptstyle B\cdot\mathrm{id}_K^\dagger} \\ \boldsymbol{K} & \xrightarrow{[\![\mathsf{read}]\!]^{\boldsymbol{K}}} & B \cdot \boldsymbol{K} \end{array} \qquad \begin{array}{cccc} K & \xrightarrow{[\![\mathsf{read}]\!]^{\boldsymbol{K}}} B \cdot K & \xrightarrow{\iota} & T_A(B \times K) \\ {\scriptstyle \mathrm{id}}\downarrow & & & \downarrow{\scriptstyle B\cdot\mathrm{id}_K^\dagger} \\ K & & \xrightarrow{[\![\mathsf{read}]\!]^{\boldsymbol{K}}} & B \cdot K \end{array}$$

which by freeness will happen just when the diagram to the right also commutes. But the map $B \cdot \mathrm{id}_K^\dagger$ therein is the quotient map by the congruence of (5.6), of which ι must be a section since it selects a family of equivalence-class representatives. □

If $\boldsymbol{K}$ is a bimodel, then $\check{\boldsymbol{K}}$ is the *maximally lazy* realisation of $\boldsymbol{K}$ as a residual comodel, wherein the program associated to each state $k \in \boldsymbol{K}$ reads the absolute minimum number of input A-tokens required to determine the next output B-token, with all subsequent reading from A handed off (via the $\mathbb{T}_A$-model structure on $\boldsymbol{K}$) to the continuation state.

Proposition 6.4. *Let $\boldsymbol{K}$ be a $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel. The image of $k \in \boldsymbol{K}$ under the unique bimodel map $\boldsymbol{K} \to \boldsymbol{E_{AB}}$ is the continuous function $\mathsf{tr}^{\check{\boldsymbol{K}}}(k)\colon A^{\mathbb{N}} \to B^{\mathbb{N}}$.*

Proof. By Lemma 6.3 we have a quotient map of bimodels $\mathrm{id}_K^\dagger \colon \boldsymbol{T}_A(\check{\boldsymbol{K}}) \twoheadrightarrow \boldsymbol{K}$ which of necessity fits into a commuting triangle

$$\begin{array}{ccc} \boldsymbol{T}_A(\check{\boldsymbol{K}}) & \xrightarrow{\mathrm{id}_K^\dagger} & \boldsymbol{K} \\ & {\scriptstyle !}\searrow \quad \swarrow{\scriptstyle !} & \\ & \boldsymbol{E_{AB}} & \end{array} .$$

The left edge of this triangle is by Proposition 6.2 the homomorphic extension of $\mathsf{tr}\check{\boldsymbol{K}} \colon K \to \mathit{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$. Thus, tracing the element $k \in K \subseteq T_A(K)$ around the two sides of this triangle yields the result. □

Finally, we give use the above results to give a comodel-theoretic reconstruction of the *reification* of each continuous function on streams by an intensional stream processor; this is the function rep_∞ of [HPG09].

Definition 6.5. The function $\mathsf{reify} \colon E_{AB} \to I_{AB}$ is the underlying map of the unique residual comodel map $\check{\boldsymbol{E}}_{\boldsymbol{AB}} \to \boldsymbol{I_{AB}}$.

As the notation suggests, we have:

Proposition 6.6. $\mathsf{reflect} \circ \mathsf{reify} = \mathrm{id}_{E_{AB}}$.

Proof. By Proposition 6.2, the unique $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel map $\boldsymbol{T}_A(\boldsymbol{I_{AB}}) \to \boldsymbol{E_{AB}}$ is $\mathsf{reflect}^\dagger$, while by Lemma 6.3, $\mathrm{id}_{E_{AB}}^\dagger$ is the unique bimodel $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel map $\boldsymbol{T}_A(\check{\boldsymbol{E}}_{\boldsymbol{AB}}) \to \boldsymbol{E_{AB}}$. So we have a (necessarily commuting) triangle of $\mathbb{T}_A$-$\mathbb{T}_B$-bimodel maps:

$$\begin{array}{ccc} \boldsymbol{T}_A(\check{\boldsymbol{E}}_{\boldsymbol{AB}}) & \xrightarrow{\boldsymbol{T}_A(\mathsf{reify})} & \boldsymbol{T}_A(\boldsymbol{I_{AB}}) \\ & {\scriptstyle \mathrm{id}_{E_{AB}}^\dagger}\searrow \quad \swarrow{\scriptstyle \mathsf{reflect}^\dagger} & \\ & \boldsymbol{E_{AB}} & \end{array}$$

and precomposing with $\eta \colon E_{AB} \to T_A(E_{AB})$ yields the result. □

This result is proved in a more general context in [Gar18, §7.3], and as explained there, the composite $\mathsf{reify} \circ \mathsf{reflect}$ implements *normalisation-by-trace-evaluation*: given an intensional stream processor, it first computes its underlying trace $A^{\mathbb{N}} \to B^{\mathbb{N}}$, and then via the reification function produces from this a maximally lazy intensional stream processor realising this trace. For instance, under this procedure, the trees $\tau_1, \tau_2 \in I_{AB}$ of Example 4.4 will both normalise to τ_1.

References

[AB20] Danel Ahman and Andrej Bauer. Runners in action. In *Programming Languages and Systems*, volume 12075 of *Lecture Notes in Computer Science*, pages 29–55. Springer, 2020. `doi:10.1007/978-3-030-44914-8_2`.

[AJ13] Mathieu Anel and André Joyal. Sweedler theory for (co)algebras and the bar-cobar constructions. Preprint, available as `arXiv:1309.6952`, 2013.

[BH96] George M. Bergman and Adam O. Hausknecht. *Co-groups and co-rings in categories of associative rings*, volume 45 of *Mathematical Surveys and Monographs*. American Mathematical Society, 1996.

[Fre66] Peter Freyd. Algebra valued functors in general and tensor products in particular. *Colloquium Mathematicum*, 14:89–106, 1966.

[Gar18] Richard Garner. Abstract hypernormalisation, and normalisation-by-trace-evaluation for generative systems. arXiv:1811.02710, 2018.

[Gar21] Richard Garner. The costructure-cosemantics adjunction for comodels for computational effects. *Mathematical Structures in Computer Science*, 2021. To appear. doi:10.1017/S0960129521000219.

[GMS20] Sergey Goncharov, Stefan Milius, and Alexandra Silva. Toward a uniform theory of effectful state machines. *ACM Transactions on Computational Logic*, 21, 2020. doi:10.1145/3372880.

[HJS07] Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3(4):4:11, 36, 2007. doi:10.2168/LMCS-3(4:11)2007.

[HPG09] Peter Hancock, Dirk Pattinson, and Neil Ghani. Representations of stream processors using nested fixed points. *Logical Methods in Computer Science*, 5:3:9, 17, 2009. doi:10.2168/LMCS-5(3:9)2009.

[HPP06] Martin Hyland, Gordon Plotkin, and John Power. Combining effects: sum and tensor. *Theoretical Computer Science*, 357:70–99, 2006. doi:10.1016/j.tcs.2006.03.013.

[KL09] Clemens Kupke and Raul Andres Leal. Characterising behavioural equivalence: three sides of one coin. In *Algebra and coalgebra in computer science*, volume 5728 of *Lecture Notes in Computer Science*, pages 97–112. Springer, 2009. doi:10.1007/978-3-642-03741-2_8.

[KRU20] Shin-ya Katsumata, Exequiel Rivas, and Tarmo Uustalu. Interaction laws of monads and comonads. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2020)*, page 15. ACM, 2020. doi:10.1145/3373718.3394808.

[Lev03] Paul Blain Levy. *Call-by-push-value*, volume 2 of *Semantic Structures in Computation*. Kluwer, 2003.

[Lin66] F. E. J. Linton. Some aspects of equational categories. In *Conference on Categorical Algebra (La Jolla, 1965)*, pages 84–94. Springer, 1966. doi:10.1007/978-3-642-99902-4_3.

[Mey75] Jean-Pierre Meyer. Induced functors on categories of algebras. *Mathematische Zeitschrift*, 142:1–14, 1975. doi:10.1007/BF01214842.

[Mog91] Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1991. doi:10.1016/0890-5401(91)90052-4.

[MS14] Rasmus Ejlers Møgelberg and Sam Staton. Linear usage of state. *Logical Methods in Computer Science*, 10:1:17, 52, 2014. doi:10.2168/LMCS-10(1:17)2014.

[PP02] Gordon Plotkin and John Power. Notions of computation determine monads. In *Foundations of software science and computation structures (Grenoble, 2002)*, volume 2303 of *Lecture Notes in Computer Science*, pages 342–356. Springer, Berlin, 2002. doi:10.1007/3-540-45931-6_24.

[PP08] Gordon Plotkin and John Power. Tensors of comodels and models for operational semantics. *Electronic Notes in Theoretical Computer Science*, 218:295–311, 2008. doi:10.1016/j.entcs.2008.10.018.

[PS04] John Power and Olha Shkaravska. From comodels to coalgebras: state and arrays. In *Proceedings of the Workshop on Coalgebraic Methods in Computer Science*, volume 106 of *Electronic Notes in Theoretical Computer Science*, pages 297–314. Elsevier, 2004. doi:10.1016/j.entcs.2004.02.041.

[PS15] Dirk Pattinson and Lutz Schröder. Sound and complete equational reasoning over comodels. In *The 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI)*, volume 319 of *Electronic Notes in Theoretical Computer Science*, pages 315–331. Elsevier, 2015. doi:10.1016/j.entcs.2015.12.019.

[PS16] Dirk Pattinson and Lutz Schröder. Program equivalence is coinductive. In *Proceedings of the 31st Annual ACM-IEEE Symposium on Logic in Computer Science (LICS 2016)*, page 10. ACM, 2016. doi:10.1145/2933575.2934506.

[PT99] John Power and Daniele Turi. A coalgebraic foundation for linear time semantics. In *CTCS '99: Conference on Category Theory and Computer Science (Edinburgh)*, volume 29 of *Electronic Notes in Theoretical Computer Science*, pages Paper No. 29020, 16. Elsevier, 1999. doi:10.1016/S1571-0661(05)80319-6.

[TW70] D. O. Tall and G. C. Wraith. Representable functors and operations on rings. *Proceedings of the London Mathematical Society*, 20:619–643, 1970. doi:10.1112/plms/s3-20.4.619.

[Uus15] Tarmo Uustalu. Stateful runners of effectful computations. In *The 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI)*, volume 319 of *Electronic*

Notes in Theoretical Computer Science, pages 403–421. Elsevier, 2015. doi:10.1016/j.entcs.2015.12.024.

[UV20] Tarmo Uustalu and Niels Voorneveld. Algebraic and coalgebraic perspectives on interaction laws. In *Programming Languages and Systems*, volume 12470 of *Lecture Notes in Computer Science*, pages 186–205. Springer, 2020. doi:10.1007/978-3-030-64437-6_10.

[vGSST90] Rob van Glabbeek, Scott A. Smolka, Bernhard Steffen, and Chris M. N. Tofts. Reactive, generative, and stratified models of probabilistic processes. In *Fifth Annual IEEE Symposium on Logic in Computer Science (Philadelphia, PA, 1990)*, pages 130–141. IEEE, 1990. doi:10.1109/LICS.1990.113740.

[Yos22] Tomoya Yoshida. Continuous functions on final comodels of free algebraic theories. In *Proceedings of MFPS 2022*, 2022. To appear.