Specifying Interaction Categories
(extended abst act)

S. Abramsky* and D. Pavlović†
Department of Computing, Imperial College, London SW7 2BZ

Abstract

We analyse two complementary methods for obtaining categorical models of process calculi. They allow adding new features respectively to the captured notion of process and to the notion of type. By alternating these two methods, all the familiar examples, as well as some new interaction categories, can be derived from basic monoidal categories.

Using the proposed constructions, interaction categories can be built and analysed without fixing a set of axioms for them. They are thus approached *via* specifications, just like algebras are approached *via* equations and relations, independantly of the intrinsic characterisation of varieties.

1 Introduction

Interaction Categories [2] are proposed as a general, yet practical tool for reasoning about functional and concurrent computation. They are not meant to be a definitive formal system, but rather a task specification, suggesting a particular framework for a solution. The paradigm of processes as relations extended in time is taken as the conceptual basis for integrating type theory with process calculus, on the background of categorical structures. The interaction of processes is captured by composition.

specifying by operations and equations; forcing is a method of specifying new models of set theory over the old ones. Note that the Birhkoff theorem, axiomatising categories that arise in universal algebra, as well as the Giraud theorem, providing the axioms for those those which arise from forcing, came only after extensive development of the corresponding specification methods. Thorough studies of the practice of specifying usually precede abstract characterisation of a class of structures.

2 Specifications and categories derived from them

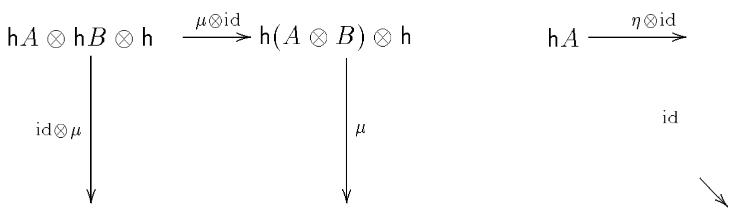
The two specification methods that we are about to describe both begin from an arbitrary, possibly degenerate interaction category $\mathcal{R}$. The first of them yields a category with the same objects as $\mathcal{R}$ but with morphisms capturing a richer notion of process, while the second one refines the type structure, but leaves the morphisms essentially unchanged.

2.1 S ecifying rocesses

Definition 2.1 *A functor* $\mathsf{h} : \mathcal{R} \to \mathcal{Q}$ *between monoidal categories* [16, sec. 1.1.] *is said to be* lax monoidal *if it is given with a natural family*

$$\begin{aligned} \mu_{AB} &: \mathsf{h}A \otimes \mathsf{h}B \longrightarrow \mathsf{h}(A \otimes B) \text{ and an arrow} \\ \eta &: \top \longrightarrow \mathsf{h}\top, \end{aligned}$$

which are coherent in the sense that for all $A, B,$ *the following diagrams commute*



cotensor $B \multimap$ in the form $(B \otimes \ ^\star)^\star$, thus making $\mathcal{R}$ autonomous (i.e. closed symmetric monoidal [16, sec. 1.5]). Now a $\star$-autonomous above, and it will be isomorphic with $\mathcal{Q}$ if and only if F is bijective on objects. In fact, any essentially surjective F induces a weak equivalence $F' : \mathcal{R}_{\mathsf{h}} \to \mathcal{Q}$, with $F = (J\,;F')$. We spell out just the 1-dimensional part of the underlying 2-adjunction. Note that it extends to $\mathcal{V}$-enriched categories for any monoidal $\mathcal{V}$ in place of Set.

Fix an autonomous $\mathcal{R}$ and consider the category $\mathcal{R}/\mathsf{Bij}$ of bijective on objects, autonomous functors out of it. A morphism from such an $F : \mathcal{R} \to \mathcal{Q}$ to $G : \mathcal{R} \to \mathcal{P}$ will be an autonomous functor $M : \mathcal{Q} \to \mathcal{P}$, satisfying $(F\,;M) = G$ (and necesssarily bijective on objects too).

On the other hand, let $[\mathcal{R}, \mathsf{Set}]_{lax\otimes}$ be the category of lax monoidal functors and lax monoidal transformations. A natural transformation $\varphi : \mathsf{h} \to \mathsf{h}'$ is said to be lax monoidal if $\eta\,;\varphi_{\top} = \eta'$ and $(\mu_{AB}\,;\varphi_{A\otimes B}) = \big((\varphi_A \times \varphi_B)\,;\mu'_{AB}\big)$.

Proposition 2.2 $\mathcal{R}/\mathsf{Bij} \quad \simeq \quad [\mathcal{R}, \mathsf{Set}]_{lax\otimes}$

2.2 S ecifying ty es

Definition 2.3 *Let $\mathcal{R}$ be a category and $\mathcal{B}$ a bicategory* [7]*. A* lax functor $\mathsf{P} : \mathcal{C} \to \mathcal{B}$ *is an assignment for each object A of $\mathcal{R}$ of an object $\mathsf{P}A$ in $\mathcal{B}$ and for each arrow $f : A \to B$ of a 1-cell $\mathsf{P}f : \mathsf{P}A \to \mathsf{P}B$ in $\mathcal{B}$. Furthermore, P comes equipped with the 2-cells*

$$\begin{array}{rcll} \mu_{fg} & : & \mathsf{P}f\,;\mathsf{P}g \longrightarrow \mathsf{P}(f\,;g) & \textit{for every composable } f \textit{ and } g, \textit{ and} \\ \eta_A & : & id_{\mathsf{P}A} \longrightarrow \mathsf{P}(\mathrm{id}_A) & \textit{for every object } A, \end{array}$$

satisfying coherence conditions similar to (1)*.*

The lax monoidal functors from 2.1 are just lax functors between monoidal categories, regarded as bicategories with one object.

Extracting from such a specification $\mathsf{P} : \mathcal{R} \to \mathsf{Rel}$ an interaction category $\mathcal{R}^{\mathsf{P}}$ is not essentially more complicated than extracting $\mathcal{R}_{\mathsf{h}}$ in 2.1, but it has very general background and deep conceptual roots.

Comprehension for categories. Consider the bicategory Span: its objects are sets, and a morphism from A to B is a pair of functions $A \leftarrow M \to B$. A 2-cell to another such pair $A \leftarrow M' \to B$ is just a function $\varphi : M \to M'$, commuting with the pairs. Given a span $B \leftarrow N \to$, the composite $A \leftarrow (M\,;N) \to$ is obtained by calculating a pullback of $M \to B$ and $B \leftarrow N$. Identities will clearly be in the form $A \stackrel{\mathrm{id}}{\leftarrow} A \stackrel{\mathrm{id}}{\to} A$. A span $A \stackrel{a}{\leftarrow} M \stackrel{b}{\to} B$ can also be viewed as an $A \times B$-matrix of sets, with $\langle a, b\rangle^{-1}(i,j)$ as the (i,j)-th entry. The 2-cells are obviously just entry-wise families of functions. The described composition then corresponds the usual matrix multiplication, using the set-theoretical sums and products.

Now any lax functor $\mathsf{P} : \mathcal{R} \to \mathsf{Span}$ induces the *total category aPhe b* between the category of functors to $\mathcal{R}$, with commutative triangles as morphisms, and the category of lax functors $\mathcal{R} \to \mathsf{Span}$ and the functional lax transformations. A lax transformation $\varphi : \mathsf{P} \to \mathsf{Q} : \mathcal{R} \to \mathsf{Span}$ is a family of matrices $\varphi_A : \mathsf{P}A \nrightarrow \mathsf{Q}A$ with a coherent 2-cell $(\mathsf{P}f\,;\varphi_B) \longrightarrow (\varphi_A\,;\mathsf{Q}f)$ for every $f : A \to B$. It is said to be *functional* if all components φ_A are functions.

The establisned equivalence extends in various directions. By dropping the functionality requirement, and varying the notion of lax transformation on the right-hand side, one gets various interesting classes of morphisms on the left-hand side: indexed profunctors and anafunctors [18], and a categorical form of simulations. On the other hand, it restricts to the Conduché correspondence [23], to the Grothendieck construction [15], and so on, until it boils down to the familiar correspondence $\mathsf{Set}/R \simeq [R, \mathsf{Set}]$ of the functions to a set R and the R-indexed sets — and, finally, to the *comprehension scheme* $\mathsf{Sub}/R \cong [R, \Omega]$, connecting the subobjects of R with the predicates over it. Indeed, just as the extension $\{x \in R | p(x)\} \hookrightarrow R$ can be obtained as a pullback of the truth $t : 1 \to \Omega$ along the predicate $p : R \to \Omega$, the construct $\int_{\mathcal{R}} \mathsf{P} \longrightarrow \mathcal{R}$ can be obtained as a pullback along $\mathsf{P} : \mathcal{R} \to \mathsf{Span}$ of the obvious projection $\mathsf{t} : \mathsf{Span}^{\bullet} \longrightarrow \mathsf{Span}$, where $\mathsf{Span}^{\bullet}$ is the total category of the identity on Span.

To restrict to the lax functors $\mathsf{P} : \mathcal{R} \to \mathsf{Rel}$, note that a relation $R \hookrightarrow A \times B$ is a jointly monic span $A \leftarrow R \to B$, i.e. a matrix of 0s and 1s. The canonical functor $\mathsf{Span} \to \mathsf{Rel}$ is

A bang thus lifts from $\mathcal{R}$ to $\mathcal{R}_{\mathsf{h}}$. However, the couniversal bang, sending each object to the corresponding cofree $\otimes$-comonoid, may loose its property in lifting.

Finally, using just definition (2), one easily shows that the (weak) products *and* coproducts are preserved and thus created by the functor $\mathcal{R} \to \mathcal{R}_{\mathsf{h}}$ as soon as the specification $\mathsf{h} : \mathcal{R} \to \mathsf{Set}$ preserves the (weak) products. However, we shall see that usually does not. Process specifications alone thus yield categories with few limits and colimits. Adding more types corrects this.

Lifting structures along type specifications is less straightforward, although quite uniform. Looking at the correspondence from proposition 2.4, one sees that any, say, binary functorial operation $\diamond$, preserved by $\int_{\mathcal{R}} \mathsf{P} \longrightarrow \mathcal{R}$, corresponds to a functional lax transformation $\mathsf{P}A \times \mathsf{P}B \stackrel{\diamond}{\longrightarrow} \mathsf{P}(A \diamond B)$, with $\langle A, \alpha\rangle \diamond \langle B, \beta\rangle = \langle A \diamond B, \alpha \diamond \beta\rangle$. In order to lift $\diamond$ from $\mathcal{R}$ to $\mathcal{R}^{\mathsf{P}}$, we must thus specify the corresponding transformations. This is where we depart from the degeneracies of $\mathcal{R}$.

3 Examples

The idea is to start from a simple model $\mathcal{R}$, and successively refine it by specifying

$$\mathcal{R} \longrightarrow \mathcal{R}_{\mathsf{h}_1} \longleftarrow (\mathcal{R}_{\mathsf{h}_1})^{\mathsf{P}_1} \longrightarrow ((\mathcal{R}_{\mathsf{h}_1})^{\mathsf{P}_1})_{\mathsf{h}_2} \longleftarrow (((\mathcal{R}_{\mathsf{h}_1})^{\mathsf{P}_1})_{\mathsf{h}_2})^{\mathsf{P}_2} \longrightarrow \cdots$$

The view of processes as relations in time suggests that any category of relations could be taken as the base $\mathcal{R}$. Namely, the calculus of relations as jointly monic spans can be developed not just over sets but over more general categories $\mathcal{C}$ [12]. The obtained category $\mathsf{Rel}(\mathcal{C})$ is always compact closed, but varying $\mathcal{C}$ allows additional structure on *actions*.

3.1 Synchrony

The simplest case is of course $\mathsf{Rel} = \mathsf{Rel}(\mathsf{Set})$. Let the process specification $\mathsf{s} : \mathsf{Rel} \to \mathsf{Set}$ assign to every set A the poset $\mathsf{s}A$ of nonempty, prefix-closed sets of finite strings from A. These strings are to be thought of as "the elements of A extended in time", so that the elements of $\mathsf{s}A$ become "the subsets of A extended in time". Algebraically, they can be presented as one-sided multiplicative systems of the free monoid A^*, i.e., the complements of the one-sided ideals of A^*.

The arrow part of s will map a relation $A \leftarrow R \to B$ to the function $\mathsf{s}R : \mathsf{s}A \to \mathsf{s}B$, defined

$$\mathsf{s}R(S) \quad = \quad \{t \in B^* | \exists s \in S. sR^*t\}, \tag{11}$$

where $A^* \leftarrow R^* \rightarrow B^*$ is the componentwise extension of R to strings. The lax monoidal structure consists of the function $\mu_{AB} : \mathsf{s}A \times \mathsf{s}B \longrightarrow \mathsf{s}(A \otimes B)$, where

$$\mu_{AB}(S,T) = \{u \in (A \otimes B)^* | \pi_A^*(u) \in S \wedge \pi_B^*(u) \in T\}, \qquad (12)$$

and $\eta \in \mathsf{s}1$ consisting of all finite strings of $\bullet \in 1$.

The category $\mathsf{sproc} = \mathsf{Rel}_\mathsf{s}$, obtained by the construction from 2.1, is a rudimentary interaction category of synchronous processes, modulo the trace equivalence. Finer notions of behaviour are obtained by taking as the elements of $\mathsf{s}A$ transition systems, or A-labelled trees, rather than just the traces $S \subseteq A^*$. Definitions (11) and (12) readily extend. Working modulo bisimilarity complicates matters [19, 20], but everything goes through.

The synchronous interaction category SProc [2] is obtained by a further type specification $\mathsf{S} : \mathsf{sproc} \rightarrow \mathsf{Rel}$. Its object part will actually be the same as for the above process specification.as

$\bullet$ must be the unit of any monoid in $\mathsf{Set}^\bullet$. Rather than $(1+A)^*$, the free monoid over $1+A$ is thus $1+A^+$, where A^+ consists of all *nonempty* strings from A.

The object part of $\mathsf{as}^\bullet$ thus takes $1+A$ to the set of prefix-closed subsets of $1+A^+$, each containing $\bullet$. The arrow part is defined using the monoid homomorphism $\widetilde{(-)} : (1+A)^* \longrightarrow 1+A^+$, which removes $\bullet$ from all nontrivial strings, and induces the weak equivalence $s \approx t \iff \widetilde{s} = \widetilde{t}$. A relation $1+A \leftarrow R \rightarrow 1+B$ now goes to the function $\mathsf{as}^\bullet R : \mathsf{as}^\bullet A \longrightarrow \mathsf{as}^\bullet B$, defined

$$\mathsf{as}^\bullet R(S) \quad = \quad \{t \in 1+B^+ | \exists s \in S.\ s \approx R^* \approx t\}. \tag{14}$$

In words, a string t belongs to $\mathsf{as}^\bullet R(S)$ if there is a string s in S such that s and t can be filled up with sequences of $\bullet$ in such a way that they become componentwise R-related.

By a similar trick, the function $\mu_{AB} : \mathsf{as}^\bullet A \times \mathsf{as}^\bullet B \longrightarrow \mathsf{as}^\bullet(A \otimes B)$ shuffles the strings:

$$\mu_{AB}(S,T) \quad = \quad \left\{u \in ((1+A)\times(1+B))^+ \,|\widetilde{\pi_A^*}(u) \in S \wedge \widetilde{\pi_B^*}(u) \in T\right\} \tag{15}$$

An element of $\mu_{AB}(S,T)$ is obtained by taking some $s \in S$ and $t \in T$, possibly of different length, interpolating $\bullet$ in them at will, to get $s' = \alpha_1 \ldots \alpha_n$ and $t' = \beta_1 \ldots \beta_n$, and then forming $u = \langle \alpha_1, \beta_1 \rangle \ldots \langle \alpha_n, \beta_n \rangle$. The unit is $\eta = \{\bullet\}$.

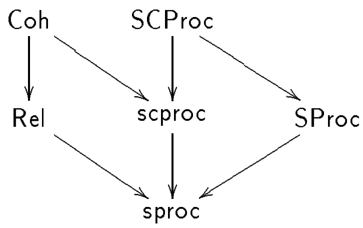
The asynchronous interaction category $\mathsf{as}^\bullet\mathsf{proc} = \mathsf{Rel}^\bullet_{\mathsf{as}^\bullet}$ is obtained as before. A version depicting a finer notion of behaviou can again obtained using $(1+A)$-labelled trees or transition systems, this time modulo weak or branching bisimilarity. A full fledged asynchronous category $\mathsf{AS}^\bullet\mathsf{Proc}$, with *weak* biproducts and a *weakly* couniversal bang, is obtained by adding more types along a specification $\mathsf{AS}^\bullet : \mathsf{as}^\bullet\mathsf{proc} \longrightarrow \mathsf{Rel}$, similar to S from section 3.1, but relaxed modulo $\approx$.

The original asynchronous category ASProc [2, sec. 5] is obtained in the same way, but using relations in place of partial functions, i.e. starting from $\mathsf{Req} = \mathsf{Rel}(\mathsf{Rel})$ rather than $\mathsf{Rel}^\bullet = \mathsf{Rel}(\mathsf{Set}^\bullet)$. Req is the category of sets and the *partial equivalence* relations on $A+B$ as the morphisms from A to B. Namely, a relation $A \nleftarrow R \nrightarrow B$ in Rel boils down to a jointly surjective pair $A \rightarrow R \leftarrow B$ in $\mathsf{Set}^\bullet$. Alternatively, Req can be viewed as the full subcategory of Rel spanned by the power sets $\wp A$. The tensor preservation along the embedding $\wp : \mathsf{Req} \rightarrow \mathsf{Rel}$ boils down to the exponential laws $\wp(A+B) \cong \wp A \times \wp B$ and $\wp 0 = 1$.

The specification $\mathsf{as} : \mathsf{Req} \rightarrow \mathsf{Set}$ assigns to each A the set of nonempty prefix closed sets of sequences from $\wp^+ A = \wp A - \emptyset$. The empty set is deleted because it plays the role of $\bullet$. The resulting category $\mathsf{asproc} = \mathsf{Req}_{\mathsf{as}}$ compares to $\mathsf{as}^\bullet\mathsf{proc}$ just as Req compares to $\mathsf{Rel}^\bullet$. For instance, bang comonads are precluded by the fact that any functor $! : \mathsf{Req} \rightarrow \mathsf{Req}$ with a natural family $e_A : !A \rightarrow 0$ must be trivial.

The structure of actions can be further enriched using other monads on Set. E.g., consider the one sending A to $1+A+A$. (If its unit is chosen to include

A in $1 + A + A$ as the first copy, then the multiplication should send the first two As from $1 + (1 + A + A) + (1 + A + A)$ to $1 + A + A$ in order, and twist the last two of them.) Besides the idling $\bullet$, this monad captures the input/output distinction — between the elements of the two copies of A. The Kleisli category $\mathsf{Set}^{\overline{\bullet}}$ for this monad can now be viewed as the category of sets, with pairs $\langle f, F\rangle$ as morphisms from A to B, where f is a partial function $A \rightharpoonup B$ and F is a subset of A. The composite of $\langle f, F\rangle : A \to B$ and $\langle g, G\rangle : B \to$ consists of the usual composite of partial function $(f\,; g)$, accompanied with the set $\left(F \cap \varphi^{-1}(G)\right) \cup \left(\overline{F} \cap \varphi^{-1}(\overline{G})\right)$, where $\overline{F}, \overline{G}$ denote the complements. The free monoid $A^{\overline{*}}$ over A in $\mathsf{Set}^{\overline{\bullet}}$ will be the quotient of $1 + (A + A)^{+}$ satisfying $\alpha\overline{\alpha} = \bullet$ for all $\alpha \in A$, with $\overline{(-)} : A + A \longrightarrow A + A$ denoting the twist map. All monoids in $\mathsf{Set}^{\overline{\bullet}}$ are thus groups — which means that any computation can be that already the synchronous ones can be specified in many different, meaningful ways.



(19)

3.4 Games

Categories of games are specifed starting from *signed* sets $\mathsf{Set}_{\pm}$. A signed set A

References

[1]

[21] D. Pavlović, Maps I: relative to a factorisation system, *J.*