Categorical logic of names and abstraction in action calculi

D. Pavlović*
COGS, University of Sussex, Brighton, UK
e-mail: duskop@cogs.susx.ac.uk

June 27, 1996

Abstract

Milner's action calculus implements abstraction in monoidal categories, so that familiar λ-calculi can be subsumed together with the π-calculus and the Petri nets. Variables are generalised to *names*: only a restricted form of substitution is allowed.

In the present paper, the well-known categorical semantics of the λ-calculus is generalised to the action calculus. A suitable functional completeness theorem for symmetric monoidal categories is proved: we determine the conditions under which the abstraction is definable. Algebraically, the distinction between the variables and the names boils down to the distinction between the transcendental and the algebraic elements. The former lead to polynomial extensions, like e.g. the ring $\mathbb{Z}[x]$, the latter to algebraic extensions like $\mathbb{Z}[\sqrt{2}]$ or $\mathbb{Z}[i]$.

Building upon the work of P. Gardner, we introduce *action categories*, and show that they are related to the static action calculus exacly as cartesian closed categories are related to the λ-calculus. Natural examples of this structure arise from allegories and cartesian bicategories. On the other hand, the free algebras for any commutative Moggi monad form an action category. The general correspondence of action calculi and Moggi monads will be worked out in a sequel to this work.

*This work was partially supported under CEC grant ERBCHBGCT930496 and under ONR grant N00014-92-J-1974.

1 Introduction

Algebraically, the λ-abstraction arises from a *property* of certain structures — namely, that each polynomial can be reduced to a normal form with a single coefficient. This property is known as *combinatorial* [5, ch. 6] or *functional* completeness [13, sec. I.6]. Having developed the algebra of variables in terms of polynomial extensions, one uses the functional completeness to define $\lambda x.\alpha(x)$ as the coefficient a of the normal form $a \cdot x$ of the polynomial $\alpha(x)$. The abstraction thus appears as an inductively derivable operation.

The idea of hiding away the variables and eliminating the substitution for the sake of function application goes back to Schönfinkel [31] and actually predates the λ-calculus. However, Schönfinkel's applicative algebras were properly understood only when Curry [4] had displayed them as the combinatorially complete kernel of the untyped λ-calculus. The abstraction seemed easier to understand as an operation, than as a property. The algebraic approach had been given a boost much later, in Lambek's categorical treatment of the functional completeness, i.e. with respect to the composition rather than the application [10]. Cartesian closed categories, as the functionally complete kernel of the typed λ-calculus, soon became an indispensable part of semantics of functional programming [3], and even induced a new algebraic interpretation of the untyped calculus [13, sec. I.17]. In a sense, the cartesian closed structure is *the* algebra of the function abstraction, and Dana Scott [32] had strongly argued that, in a conceptual world, it should have been discovered *before* the λ-calculus itself.

However, beyond the realm of functions, the notion of abstraction is far less clear cut. And the realm of functions does not suffice for studying computations, since they may yield no output, or may yield several outputs, as soon as nondeterminism or concurrency enter scene. Moggi's computational monads [23] measure the deviation from functionality which comes with various notions of computation, while Abramsky speaks of processes as *relations extended in time* [1]. In more than one way, the step from functions to computations echoes the step to relations. One of the most conspicuous structural features in the world of computations, as well as in the world of relations, is a tensor product, induced by what used to be the cartesian product in the old world of functions. Quite different models share it: Petri nets [24], Chu spaces [30], action structures [18], interaction categories [1]... — they all carry this monoidal structure, roughly corresponding to the parallel composition. A "simple type theory" of processes seems to be emerging.

In this monoidal world, the abstraction arises in connection with parametrised processes, just like the λ-abstraction had arisen to bind parametrised families of functions. As it is well known, the dependancy of a process on the values of data can be eliminated, i.e. reduced to a choice between atomic actions [17, sec. 2.8]. However, when processes are allowed to communicate to each other names of communication channels as parameters, a genuinely new kind of situation arises. The structure of a process depending on such a parameter may change during the execution, since different communication channels that may be received can open the different computation paths, or preempt them. This is the idea of a *mobile* process. The corresponding generalisation of the λ-calculus is the π-calculus [22]. The main difference is that a function is *applied* to its input sequentially, while a mobile process can *communicate* with any of the processes running in parallel with it, provided that there is a common communication channel. The function application has been generalised to the communication, which may be non-sequential and non-deterministic.

The main feature of the π-abstraction is that it does not just bind a parameter x, for which the received input is to be substituted, like the λ-abstraction does, but it also specifies a communication channel y, where the sought input must be received. A π-abstraction operator is thus in the form $y(x)$, and it binds x, and adds y as a free channel name, ready for the input. A process prefixed with such an operator will not consume just any argument immediately preceding it, and substitute it for x, like an old λ-term would do. A process/π-term must first find a parallel process (another π-term) sending its output through y. It can be recognized by a prefix in the form $\overline{y}u$, where u is the name being sent. In case several such are running in parallel, prefixed, say, with $\overline{y}u_1$, $\overline{y}u_2$ etc., only one of their outputs/prefixes will be consumed, and the choice will be made nondeterministically. Any of the channels $u_1, u_2 \ldots$ may thus end up being substituted for x.

The *action calculus* provides means for reducing this complex reduction procedure to a familiar abstraction/application routine. There are two dimensions of this apparent syntactic miracle: the *controls*, and the *dynamics*. The controls mark the input and the output channels, and block the reduction unless these coincide. The dynamics is the reduction preorder that, unlike the Church-Rosser situations, cannot be hidden away from semantics. Indeed, upon different communications, processes may reduce in essentially different ways.

However, in the present paper, we shall neglect the dynamic side, and try to show that the remaining *static action calculus* is a kind of "monoidal λ-calculus" plus controls. The dynamics of the full-blown structure is undoubtedly essential for its computational purposes, but the underlying abstraction mechanism seems to be independant on this superstructure.

Peculiar as it may appear at the first sight (due to its unusual mixture of categorical and syntactical features) the action calculus can be formalised as a generalisation of Lambek's κ-calculus [13, sec. I.6], the first-order version of the typed λ-calculus. It is extended in two directions: by adding certain graphic operations, the mentioned controls, and by weakening the cartesian setting to a special monoidal structure. In section 2, we extend the existing theory of the functional completeness for cartesian (closed) categories [13, part I] in these two directions. In section 3, *action categories* are introduced, and shown to generate the action calculus in a similar way as cartesian closed categories generate the λ-calculus: the action abstraction arises from their functional completeness. This provides a base for categorical semantics of the action calculus.

Several classes of natural examples, modelling static action calculi, are described in subsection 3.3. A different application of the obtained semantics is presented in subsection 3.2: it is shown how allowing the ordinary substitution reduces the action calculus to the cartesian κ-calculus. This sheds some light on the degeneration of the extensional higher-order action calculus to the ordinary λ-calculus, described in [20] — and shows how narrow is the passage from the cartesian to the monoidal abstraction. The simplicity with which the latter has been introduced in the action calculus conceals a genuinely fundamental idea, mostly behind the concept of a name, with its constrained substitution.

While weakening the cartesian setting leaves the abstraction operations virtually unchanged, it has deep repercussions on the substitution, which on its turn weakens the β-reduction. The original constraints, imposed on the substitution in the action calculus, were computationally motivated: if names are channel parameters, then only the proper channel names should be substituted for them, and surely not arbitrary process expressions. Variables, as the value parameters, on the other hand, accomodate substitution of all expressions that can be evaluated.

The algebraic treatment provides different explanations. In algebra, the substitution is implemented by means of extensions: a variable x, freely adjoined to, say, the ring of integers $\mathbb{Z}$, leads to the polynomial extension $\mathbb{Z}[x]$. The fact that x is free for substitution of any element means that it is *transcendental*, unconstrained by any equations over $\mathbb{Z}$. On the other hand, an *algebraic* element, which does satisfy some equations, can only be replaced by elements satisfying the same equations. E.g., if $x^2 - 2 = 0$ holds for x, then only $-x$ can be substituted for it, without invalidating the equation. Of course, this x is just $\sqrt{2}$ and the substitution constraint formally means that the extension $\mathbb{Z}[\sqrt{2}] = \mathbb{Z}[x]/(x^2 - 2)$ has exactly one nontrivial endomorphism fixing $\mathbb{Z}$, induced by the assignment $\sqrt{2} \mapsto -\sqrt{2}$. Conversely, the "name" $\sqrt{2}$ can be viewed as an abbreviation of something like $[x; x^2 = 2]$. We shall later present names exactly in this form.

The idea that names are some algebraic elements, as opposed to variables as transcendental elements, suggests a general treatment of the constrained substitution, along the lines of Galois theory. Luckily, this general treatment need not be developed very far: the names arising in the action calculus turn out to be the algebraic elements of a very special kind, characterised by a simple set of equations (23–25). They are consequences of the conversion rules of the name abstraction (cf. definition 3.2). The point of the functional completeness is that the name abstraction, together with its conversion rules, can also be derived as a consequence of the equations determining the names.

The basic category theory, to the extent of the first 30 pages of [13] is assumed to be familiar to the reader. The next 50 pages of that book provide a succinct exposition of the main ideas of categorical semantics, presently applied to the action calculus. In order to align the presented constructions with that standard material, I deviated from some of the action calculus notation. In particular, the composition is written in the form $f \circ g$, (referring to $(f \circ g)(x) = f(g(x))$) rather than $g \cdot f$. The connection with the prefixing operation, which motivates this latter notation, is not yet considered here.

2 κ-calculus

2.1 Graph algebra

The elements of universal algebra over the category of graphs have been outlined in [13]. The basic feature is that the operations take arrows as arguments. The arity is thus not just a number, but may involve some equations imposed on the sources and the targets. It is convenient to present such arities as *deductive systems*, with Gentzen-style derivation rules. For instance, the pairing and the currying can be respectively introduced:

$$\frac{f : k \to m \qquad g : k \to n}{\langle f, g\rangle : k \longrightarrow m \times n} \qquad\qquad \frac{h : k \times m \longrightarrow n}{h^* : k \longrightarrow n^m}$$

Of course, to be able to recover f and g from $\langle f, g\rangle$, one needs the composition and the projections, together with the equations to tie up the cartesian structure [13, sec. I.3]; and to uncurry h^*, one also needs the closed structure [*ibidem*].

In many cases, a deductive system can be reduced to a purely equational theory in terms of functors and natural transformations. For instance, the cartesian structure can be given by a symmetric tensor product $\otimes$ with a unit $\top$ and natural transformations $\Delta : k \to k \otimes k$ and $\omega : k \to \top$, making each k into a commutative comonoid:

$$\begin{array}{rcccl} (\omega \otimes k) \circ \Delta & = & \mathrm{id}_k & = & (k \otimes \omega) \circ \Delta \qquad (1) \\ \Delta \circ (\Delta \otimes k) & & = & & \Delta \circ (k \otimes \Delta) \qquad (2) \\ \tau \circ \Delta & & = & & \Delta \qquad (3) \end{array}$$

where $\tau : k \otimes k \xrightarrow{\sim} k \otimes k$ is a component of the tensor symmetry. While such algebraic presentations tend to be more succinct, deductive systems are established as a more versatile tool for practical tasks.

Definition 2.1 *A* graphic signature *is a set $\mathcal{K}$ of operation symbols, here generically denoted by C, each of them given with an arity rule*

$$\frac{a_1 : m_1 \to n_1 \qquad a_2 : m_2 \to n_2 \qquad \cdots \qquad a_r : m_r \to n_r}{C(a_1, \ldots, a_r) : m \longrightarrow n} \qquad (4)$$

Together with a set of well-formed equations, such a signature presents a graphic theory *$\mathcal{K}$.*

The notion of a model is defined in the usual way. For each symbol $C \in \mathcal{K}$, a $\mathcal{K}$-graph $\mathbb{A}$ comes equipped with a distinguished partial mapping $C : \mathbb{A}^r \rightharpoonup \mathbb{A}$, satisfying whatever constraints may have been imposed.

In the present paper, we shall only be concerned with the theories that extend *symmetric monoidal categories* [15, sec. VII.1]. Even if not mentioned, the symmetry will be *always assumed.* Fix an arbitrary graphic theory $\mathcal{K}$ and add the composition $\circ$ with the identities id, and a symmetric tensor $\otimes$ with the unit $\top$. The resulting graphic theory will be denoted $\mathcal{K}_\otimes$. For simplicity, we assume that the monoidal structure is strictly associative and unitary. A $\mathcal{K}_\otimes$*-category* $\mathbb{A}$ is thus a commutative monoid in Cat, equipped with the $\mathcal{K}$-operations. Since every monoidal category is equivalent to a strict one, this strictness assumption causes no loss of generality.

2.2 Extensions

Given a $\mathcal{K}_\otimes$-category $\mathbb{A}$, we want to extend it by a set of formal arrows $X = \{x, y, z \ldots\}$ from $\top$ to various objects of $\mathbb{A}$. X can thus be viewed as a set symbols with a function *type* : $X \rightarrow \mathbb{A}$, or as a multiset of objects from $\mathbb{A}$. Furthermore, a set Q of well-formed equations in the elements of X and $\mathcal{K}_\otimes$ may be imposed. The resulting extension will be denoted $\mathbb{A}[X;Q]$. The pair $[X;Q]$ is a set of *names.* The extension $\mathbb{A}[X;Q]$ is the smallest $\mathcal{K}_\otimes$-category generated by $\mathbb{A}$ and $[X;Q]$. It is formed in the following stages:

- for each $x \in X$ of type $k \in \mathbb{A}$, add to the graph $\mathbb{A}$ an edge $x : \top \rightarrow k$;
- close the resulting graph under the $\mathcal{K}$-operations and
- generate a $\otimes$-category; finally
- enforce the Q-equations.

Clearly, the objects remain unchanged: the obvious functor $\mathsf{ad} = \mathsf{ad}_{[X;Q]} : \mathbb{A} \longrightarrow \mathbb{A}[X;Q]$ is identity on objects. It is furthermore universal among the $\mathcal{K}_\otimes$-functors from $\mathbb{A}$ to the categories with an interpretation of $[X;Q]$.

Conventions. The arrows of an extension $\mathbb{A}[X;Q]$ will be denoted by $\alpha, \beta, \ldots, \varphi$, while the arrows of $\mathbb{A}$ remain $a, b, \ldots, f$. The objects of both categories are k, ℓ, m, n.

As usually in the polynomial notation, instead of, say, $\mathbb{A}[\{x\} \cup Y \cup Z; Q]$, for $x \in X$ and $Y, Z \subseteq X$, we simply write $\mathbb{A}[x, Y, Z; Q]$.

2.3 Functional completeness

Proposition 2.2 *Let $\mathbb{A}$ be a $\mathcal{K}_\otimes$-category and $\mathbb{A}[x;Q]$ its extension by a name $[x;Q]$ of type $k \in \mathbb{A}$. The following conditions are equivalent:*

(a) *For each $\alpha : m \longrightarrow n$ in $\mathbb{A}[x;Q]$ there is a unique $\kappa x.\alpha : k \otimes m \longrightarrow n$ in $\mathbb{A}$, such that*

$$\alpha \quad = \quad \mathsf{ad}(\kappa x.\alpha) \circ (x \otimes m) \tag{5}$$

(b) *The functor $\mathsf{ad} : \mathbb{A} \longrightarrow \mathbb{A}[x;Q]$ has a left adjoint ab, such that the composite $\mathsf{ab} \circ \mathsf{ad}$ is just tensoring with k. The unit and the counit of the adjunction are respectively in the forms $\eta_m = x \otimes m$ and $\varepsilon_m = \omega \otimes m$, for some $\omega : k \to \top$.*

(c) *$\mathbb{A}[x;Q]$ is isomorphic with the Kleisli category for a comonad over the endofunctor $k \otimes (-) : \mathbb{A} \longrightarrow \mathbb{A}$.*

Proof. **(a)$\Rightarrow$(b)** Consider the maps

$$\mathbb{A}[x;Q]\Big(m, n\Big) \underset{\mathsf{ad}(-)\circ(x\otimes m)}{\overset{\kappa x.(-)}{\rightleftarrows}} \mathbb{A}\Big(k \otimes m, n\Big) \tag{6}$$

Condition (5) says that going to the right and back yields the same arrow. The other way around, take any $g \in \mathbb{A}(k \otimes m, n)$ and apply (5) to $\alpha = \mathsf{ad}(g) \circ (x \otimes m)$. Hence

$$\mathsf{ad}(g) \circ (x \otimes m) \quad = \quad \mathsf{ad}\Big(\kappa x.\mathsf{ad}(g) \circ (x \otimes m)\Big) \circ (x \otimes m). \tag{7}$$

The uniqueness part of (a) now gives

$$g \quad = \quad \kappa x.\mathsf{ad}(g) \circ (x \otimes m). \tag{8}$$

In other words, going on (6) to the left and back yields the same arrow again. (6) is a bijection.

The required adjunction is now obtained by extending this bijection to a *natural* isomorphism

$$\mathbb{A}[x;Q]\Big(m,\ \mathsf{ad}(n)\Big) \quad \cong \quad \mathbb{A}\Big(\mathsf{ab}(m),\ n\Big). \tag{9}$$

The object part of $\mathsf{ab} : \mathbb{A}[x;Q] \longrightarrow \mathbb{A}$ clearly maps $m \longmapsto k \otimes m$. The requirement that (9) be natural in m determines that $\varphi \in \mathbb{A}[x;Q](m', m)$ must be mapped to

$$\mathsf{ab}(\varphi) \quad = \quad \kappa x.(x \otimes m) \circ \varphi, \tag{10}$$

which is the only arrow making the square

$$\begin{tikzcd}
m' \arrow[r, "x\otimes m'"] \arrow[dd, "\varphi"'] & k \otimes m' \arrow[dd, "\mathsf{ad}\circ\mathsf{ab}(\varphi)"'] \arrow[dr, "\mathsf{ad}(\kappa x.\alpha\circ\varphi)"] & \\
 & & n \\
m \arrow[r, "x\otimes m"'] & k \otimes m \arrow[ur, "\mathsf{ad}(\kappa x.\alpha)"'] &
\end{tikzcd} \tag{11}$$

commute. In the presence of (a), this commutativity is equivalent to the equation $\kappa x.\alpha \circ \varphi = (\kappa x.\alpha) \circ \mathsf{ab}(\varphi)$. This is the required naturality of (9) in m.

The naturality in n boils down to the equation $\kappa x.\mathsf{ad}(f) \circ \alpha = f \circ \kappa x.\alpha$, for all $f \in \mathbb{A}(n, n')$. But this is again a consequence of the uniqueness part of (a).

Hence the adjunction $\mathsf{ab} \dashv \mathsf{ad}$. By correspondence (6) its unit and counit will be as asserted in (b). So it remains to prove that $\mathsf{ab} \circ \mathsf{ad}(f)$ is $k \otimes f$ for all f from $\mathbb{A}$.

Since $\mathbb{A}[x; Q]$ is a monoidal category, any $\varphi : m' \to m$ in it satisfies

$$(x \otimes m) \circ \varphi = x \otimes \varphi = (k \otimes \varphi) \circ (x \otimes m'). \tag{12}$$

Putting $\varphi = \mathsf{ad}(f)$, we get that

$$\begin{tikzcd} m' \arrow[r, "x\otimes m'"] \arrow[d, "\mathsf{ad}(f)"'] & k \otimes m' \arrow[d, "\mathsf{ad}(k\otimes f)"] \\ m \arrow[r, "x\otimes m"'] & k \otimes m \end{tikzcd} \tag{13}$$

commutes, since ad is a monoidal functor, identity on objects, and hence $k \otimes \mathsf{ad}(f) = \mathsf{ad}(k \otimes f)$. But (13), together with the uniqueness part of (a), implies that $k \otimes f$ is equal to $\kappa x.(x \otimes m) \circ \mathsf{ad}(f)$, which is just $\mathsf{ab} \circ \mathsf{ad}(f)$, by definition (10).

(b)⇒(c) The adjunction $\mathsf{ab} \dashv \mathsf{ad} : \mathbb{A} \longrightarrow \mathbb{A}[x; Q]$ induces a comonad in the standard way [13, prop. I.6.2]. The isomorphism of $\mathbb{A}[x; Q]$ and the Kleisli category $\mathbb{A}_{k\otimes}$ is identity on objects, while its arrow part is given by the mapping κx from (6). Indeed, the morphisms of $\mathbb{A}_{k\otimes}$ are detemined exactly as to make κx functorial.[1]

(c)⇒(a) When $\mathbb{A}[x; Q]$ is transformed along the given isomorphism into $\mathbb{A}_{k\otimes}$, the mapping κx becomes the presentation of the Kleisli arrows as they appear in $\mathbb{A}$. Viewed in $\mathbb{A}_{k\otimes}$, the arrow $x \otimes m$ becomes the identity on $k \otimes m$, while the functor $\mathsf{ad} : \mathbb{A} \longrightarrow \mathbb{A}_{k\otimes}$ becomes the cofree one: it is identity on objects, and sends $a : m \longrightarrow n$ to $\omega \otimes a : k \otimes m \longrightarrow n$.

With the described data, and the Kleisli composition, equation (5) easily follows. □

Definition 2.3 *A $\mathcal{K}$-category $\mathbb{A}$ is* functionally complete *with respect to a name $[x; Q]$ if either of the equivalent conditions of proposition 2.2 is satisfied. The*

[1]For a more abstract proof, note that that the Kleisli category $\mathbb{A}_G$ for any comonad $G : \mathbb{A} \to \mathbb{A}$ can be characterised, up to isomorphism, by the existence of a functor $I : \mathbb{A} \to \mathbb{A}_G$, which is identity on objects and has a left adjoint.

functor $\mathsf{ad} : \mathbb{A} \longrightarrow \mathbb{A}[x;Q]$ adds a dummy x, *whereas* $\mathsf{ab} : \mathbb{A}[x;Q] \longrightarrow \mathbb{A}$ abstracts *over* x.

$\mathbb{A}$ *is functionally complete with respect to a set of names* $[X;Q]$ *if for all* $x \in X$ *and* $Y \subseteq X \setminus \{x\}$, $\mathbb{A}[Y;Q]$ *is functionally complete with respect to* $[x;Q]$. *Moreover, the abstraction should be uniform, in the sense that the diagram*

$$\begin{tikzcd}
\mathbb{A}[x,Y;Q] \arrow[r, "\mathsf{ab}_x^Y"] \arrow[d, "\mathsf{ad}_Z^{x,Y}"'] & \mathbb{A}[Y;Q] \arrow[d, "\mathsf{ad}_Z^Y"] \\
\mathbb{A}[x,Y,Z;Q] \arrow[r, "\mathsf{ab}_x^{Y,Z}"'] & \mathbb{A}[Y,Z;Q]
\end{tikzcd} \tag{14}$$

must commute for all $Y, Z \subseteq X \setminus \{x\}$. *Moreover, the canonical isomorphism*

$$\mathsf{ab}_y^Z \circ \mathsf{ab}_x^{y,Z} \cong \mathsf{ab}_x^Z \circ \mathsf{ab}_y^{x,Z} \quad : \quad \mathbb{A}[x,y,Z;Q] \longrightarrow \mathbb{A}[Z;Q] \tag{15}$$

induced by the fact that both sides are adjoint to $\mathsf{ad}_{x,y}^Z$, *must come from the symmetry* $\tau : k \otimes \ell \xrightarrow{\sim} \ell \otimes k$.

An abstraction situation *is a pair* $(\mathcal{K}, Q)$, *such that every* $\mathcal{K}_\otimes$*-category* $\mathbb{A}$ *is functionally complete with respect to any set of names* $[X;Q]$.

Remarks. This last quantifier over all sets of names is not as extensive as it appears: by proposition 2.2, $\mathbb{A}$ is functionally complete for $[X;Q]$ if and only it is functionally complete for $[X';Q]$, where X' is the image of *type* $: X \to \mathbb{A}$. To check an abstraction situation, one only needs to consider the subsets X' of $|\mathbb{A}|$ (but still for all $\mathcal{K}_\otimes$-categories $\mathbb{A}$, though).

The notion of functional completeness, as defined above, should not be confused with its homonym in duality theory [12], where, say, the boolean algebra 2 is functionally complete because every function $2^n \to 2$ corresponds to a polynomial. In boolean algebras, there are thus enough polynomials to represent all functions, whereas we are concerned with the situations when there are enough constants to represent all polynomials. The term polynomial completeness might be better, but the usage, at least for the cartesian case, seems completely standard.

2.4 Characterising abstraction

Commutative comonoids and cartesianness. Let $\mathbb{N}$ be the full subcategory of Set^{op} spanned by the natural numbers: a morphism $m \to n$ in $\mathbb{N}$ is just a function $m \leftarrow n$. Since Set is the coproduct completion of 1, Set^{op} is the product completion [14]. The free cartesian category generated by 1 is thus $\mathsf{Set}_{\text{fin}}^{op}$, and

$\mathbb{N}$ is the free *strictly* cartesian category over 1, with $+$ as the cartesian product and 0 as the terminal object.

Lemma 2.4 *For every object k of a strict monoidal category $\mathbb{A}$, there are bijective correspondences between*

(a) *the monoidal functors $\mathbb{N} \longrightarrow \mathbb{A}$ mapping $1 \mapsto k$,*

(b) *the commutative comonad structures on the functor $k \otimes (-) : \mathbb{A} \longrightarrow \mathbb{A}$,*

(c) *the commutative comonoid structures on k.*

As mentioned before, a commutative comonoid structure, dual to a commutative monoid, is a pair $\top \overset{\omega}{\longleftarrow} k \overset{\Delta}{\longrightarrow} k \otimes k$, satisfying (1–3). A comonoid homomorphism is an arrow u making the following diagram commute.

$$\begin{tikzcd} & \ell \arrow[dl, "\omega"'] \arrow[r, "\Delta"] \arrow[d, "u"'] & \ell \otimes \ell \arrow[d, "u\otimes u"] \\ \top & k \arrow[l, "\omega"] \arrow[r, "\Delta"'] & k \otimes k \end{tikzcd} \tag{16}$$

The general facts about the commutative (co)monads can be found in [9]. In the particular case of the comonad $k \otimes (-)$, though, the commutativity is probably best understood simply as defined by the above lemma: it really boils down to the commutativity of the corresponding comonoid. In the sequel, *all comonoids and comonads are commutative*, even when this is not emphasized.

By proposition 2.2(c), the equivalent structures from the above lemma are necessary for functional completeness: they ensure the existence of the Kleisli category $\mathbb{A}_{k\otimes}$. This category is monoidal if and only if the comonad $k \otimes (-)$ is commutative. The abstraction situations thus require that all objects are commutative comonoids. The arrows do not have to be comonoid homomorphisms, though. Hence the non-cartesian abstraction situations, properly extending the existing categorical theory of abstraction [13, 3]. However, the adjoined names do have to be comonoid homomorphisms: the isomorphism $\mathbb{A}[x; Q] \cong \mathbb{A}_{k\otimes}$ depends on that.

The situations in which all objects are comonoids but some arrows are not comonoid homomorphisms are ubiquitous. E.g., the category Rel of sets and relations, with the tensor induced by the cartesian product from Set, inherits the comonoid structures $1 \leftarrow k \rightarrow k \times k$, but the set-theoretical diagonals and the terminal functions to 1 do not form natural families with respect to all relations. Indeed, if diagram (16) is taken in Rel, the commutativity of the square means that u is a single-valued relation; and the commutativity of the triangle — that it is total. The comonoid homomorphisms in Rel are just the functional relations.

Admissibility and controls. To ensure that the $\mathcal{K}_\otimes$-structure extends from $\mathbb{A}$ to $\mathbb{A}_{k\otimes}$, some additional conditions are needed. A type theoretical version is in [7, def. 3.6], categorical in [8, def. 8.1(4)] and [28, def. 4.1]. The following is closer to the former.

Definition 2.5 *Let C_ℓ be a graphic operation with the arity*

$$\frac{b_1 : \ell \otimes m_1 \to n_1 \qquad b_2 : \ell \otimes m_2 \to n_2 \qquad \cdots \qquad b_r : \ell \otimes m_r \to n_r}{C_\ell(b_1, \ldots, b_r) : \ell \otimes m \longrightarrow n} \tag{17}$$

Given two such, C_k and C_ℓ, an arrow $u : \ell \to k$ in $\mathbb{A}$ is said to be admissible *if it satisfies*

$$C_\ell\Big(b_1(u \otimes m_1), \ldots, b_r(u \otimes m_r)\Big) \quad = \quad C_k(b_1, \ldots, b_r) \circ (u \otimes m). \tag{18}$$

A monoidal functor $\mathbb{M} \to \mathbb{A}$ is admissible if its image consists of arrows admissible with respect to a given family $\{C_\ell\}_{\ell \in \mathbb{M}}$.

*Finally, a comonoid k in a $\mathcal{K}_\otimes$-category $\mathbb{A}$ is admissible if every graphic operation $C \in \mathcal{K}$ induces a unique family making the monoidal functor $\mathbb{N} \to \mathbb{A} : 1 \mapsto k$ admissible. Such an operation is called k-*control.

Each morphism $m \leftarrow n$ of $\mathbb{N}$ decomposes as $m \hookleftarrow m' \simeq n' \twoheadleftarrow n$, where the first and the last components are monotone. It is not hard to see that a monoidal functor $\mathbb{N} \to \mathbb{A}$ must take every monotone injection $m \hookleftarrow m'$ to an arrow derived from ω and $\otimes$, every monotone surjection $n' \twoheadleftarrow n$ to an arrow derived from Δ and ω and every bijection $m' \simeq n'$ to a composite of the symmetries τ. Since the class of arrows satisfying (18) is clearly closed under the composition, checking whether $\mathbb{N} \to \mathbb{A}$ is admissible boils down to checking separately the admissibility of the arrows derived from ω and $\otimes$, from Δ and $\otimes$, and from τ, $\otimes$ and $\circ$. These three parts correspond to conditions 1–3 from [7, def. 3.6].

On the other hand, every tensor power k^j of a commutative comonoid k is a commutative comonoid again. Hence the Kleisli categories $\mathbb{A}_{k^j\otimes}$ for all natural numbers j. The mapping $j \mapsto \mathbb{A}_{k^j\otimes}$ determines an indexed category $\mathbb{N}^{op} \to \mathsf{Cat}$, with the reindexing induced by the precomposition. Condition (18) now appears as the naturality with respect to this reindexing, and a k-control is just an indexed graph operation on this indexed category. The setting described in [8, def. 8.1(4)] and [28, def. 4.1] is built upon this idea[2].

Proposition 2.6 *A $\mathcal{K}_\otimes$-category $\mathbb{A}$ is functionally complete with respect to the extension by $[x; Q]$ of type k if and only if*

(i) *k is an admissible commutative comonoid in $\mathbb{A}$, and*

(ii) *x is an admissible comonoid homomorphism in $\mathbb{A}[x; Q]$.*

[2]However, it seems that these graphic operations as natural transformations must be total and monotone with respect to the reduction preorder, so that, e.g., currying, or replication [16] cannot be treated directly.

Proof. When $\mathbb{A}$ is functionally complete, the commutative comonoid structure $\top \xleftarrow{\omega} k \xrightarrow{\Delta} k \otimes k$ is

$$\begin{aligned} \omega &= \kappa x.\mathrm{id}_\top && (19) \\ \Delta &= \kappa x.x \otimes x. && (20) \end{aligned}$$

Its admissibility is proved as in [7, sec. 4.2].

The fact that x is an admissible comonoid homomorphism from $\top \xleftarrow{\mathrm{id}} \top \xrightarrow{\mathrm{id}} \top \otimes \top$ follows directly from (5).

The other way around, assume (i) and (ii). The abstraction κx is defined inductively. An arrow of $\mathbb{A}[x;Q]$ is either some a from $\mathbb{A}$, or x itself, or a composite of previously generated $\mathbb{A}[x;Q]$-arrows, or their tensor. Finally, it may be obtained using some $C \in \mathcal{K}$ as the outermost operation.

The base cases

$$\begin{aligned} \kappa x.a &= \omega \otimes a \\ \kappa x.x &= \mathrm{id}_k \end{aligned}$$

and the step cases

$$\begin{tikzcd} k \otimes m \arrow[r, "\kappa x.\beta \circ \alpha"] \arrow[d, "\Delta \otimes m"'] & p \\ k \otimes k \otimes m \arrow[r, "k \otimes \kappa x.\alpha"'] & k \otimes n \arrow[u, "\kappa x.\beta"'] \end{tikzcd}$$

$$\begin{tikzcd} k \otimes m \otimes m' \arrow[r, "\kappa x.\alpha \otimes \alpha'"] \arrow[d, "\Delta \otimes m \otimes m'"'] & n \otimes n' \\ k \otimes k \otimes m \otimes m' \arrow[r, "k \otimes c \otimes m'"'] & k \otimes m \otimes k \otimes m' \arrow[u, "(\kappa x.\alpha) \otimes (\kappa x.\alpha')"'] \end{tikzcd}$$

are standard [13, prop. I.2.1], and only depend on the assumption that k is a commutative comonoid. The admissibility assumption is needed for

$$\kappa x.C(\alpha_1, \ldots, \alpha_r) \quad = \quad C_k(\kappa x.\alpha_1, \ldots, \kappa x.\alpha_r)$$

This step is the main contribution of [7]. The soundness of the resulting κx abstractor is proved in [7, theorem 4.6].

Condition (ii) ensures the validity of (5): an inductive argument suffices again. The admissibility of x yields

$$\mathsf{ad}\Big(\kappa x.C(\alpha_1, \ldots, \alpha_r)\Big) \circ (x \otimes m) \quad = \quad \mathsf{ad}\Big(C_k(\kappa x.\alpha_1, \ldots\Big) \circ (x \otimes m)$$

$$\begin{aligned} &= C_k\Big(\mathsf{ad}(\kappa x.\alpha_1), \ldots\Big) \circ (x \otimes m) \\ &= C\Big(\mathsf{ad}(\kappa x.\alpha_1) \circ (x \otimes m), \ldots\Big) \\ &= C(\alpha_1, \ldots, \alpha_r). \end{aligned}$$

The remaining cases only depend on x being a comonoid homomorphism. The uniqueness part of 2.2(a) follows by a similar inductive argument. $\square$

Corollary 2.7 *$(\mathcal{K}, Q)$ is an abstraction situation if and only if*

(i) *$\mathcal{K}$ makes all objects into admissible commutative comonoids, while*

(ii) *Q makes all names into admissible comonoid homomorphisms.*

3 Action calculi and control structures

3.1 Abstraction elimination

Definition 3.1 *A monoidal category where every object has a commutative comonoid structure is said to be* semi-cartesian*.*

An action *category is a $\mathcal{K}_\otimes$-category with a distinguished admissible commutative comonoid structure on every object.*

A semi-cartesian category is cartesian if and only if each object carries a unique comonoid structure, and such structures form two natural families, Δ and ω. The naturality means that all morphisms of the category must be comonoid homomorphisms.

In action categories, the *property* of semi-cartesianness is fixed as *structure*: on each object, a particular comonoid structure is chosen. This choice may be constrained by some given graphic operations, with respect to which the structures must be admissible. The proof of proposition 2.6 shows that such structures determine the abstraction operators, and are determined by them. This is the essence of the equivalence of action categories and action calculi.

As the embodiment of 2.7(i), action categories satisfy

$$\frac{\text{action calculi}}{\text{action categories}} = \frac{\text{typed } \lambda\text{-calculi}}{\text{cartesian closed categories}} \tag{21}$$

On both sides, the semantics of the numerator is developed using extensions of the denominator: the polynomial ones on the right hand side, and the algebraic ones, satisfying 2.7(ii), on the left. The former can actually be viewed as a special case of the latter, since cartesian categories subsume under action categories, while the λ-calculus is isomorphic to the corresponding special case of the κ-calculus [13, sec. I.6]. In a sense, condition 2.7(ii) is thus the "value" of fraction (21). To give a precise meaning to this idea, we need a formal definition of the action calculus.

Definition 3.2 [19, 16] *Given a $\mathcal{K}_\otimes$-category $\mathbb{A}$ and a multiset X over A, let $\mathbb{A}[X;\mathsf{ab}]$ be the smallest $\mathcal{K}_\otimes$-category obtained by extending $\mathbb{A}$ with*

- *$\omega : k \to \top$ for every k, and*
- *$x : \top \to k$ for each $x \in X$ of type k,*

and then closing the obtained $\mathcal{K}_\otimes$-category under the operation

$$\frac{\alpha : m \to n}{\mathsf{ab}_x\alpha : k \otimes m \longrightarrow k \otimes n} \tag{22}$$

which is required to be functorial and further to satisfy

$$\begin{array}{rrcll}
\sigma : & (\kappa x.\alpha) \circ (y \otimes m) & = & \alpha(y/x) & (\alpha : m \to n) \\
\delta : & \kappa x.x \otimes m & = & \mathrm{id}_{k\otimes m} & (x : \top \to k) \\
\gamma : & \kappa x.a & = & \omega \otimes a & (x \notin \mathbf{fn}(a)) \\
\zeta : & (\alpha \otimes \beta) \circ p_{km} & = & p_{\ell n} \circ (\beta \otimes \alpha) & (\alpha : m \to n, \beta : k \to \ell)
\end{array}$$

where $\kappa x.\varphi$ abbreviates $(\omega \otimes n) \circ \mathsf{ab}_x\varphi$ and $p_{k\ell}$ is $\kappa xy.y \otimes x$ for $x : \top \to k$ and $y : \top \to \ell$. (The inductive definition of the set $\mathbf{fn}(\alpha)$ of free names of α is as usually, with ab_x binding x.)

The resulting $\mathcal{K}_\otimes$-category $\mathbb{A}[X;\mathsf{ab}]$ is a (static) action calculus*.*

The *dynamic* part of the action calculus is a preorder enrichment of the category $\mathbb{A}[X;\mathsf{ab}]$. This *reduction* preorder is expanded from a set of reduction rules, taking care that the free names are preserved, and that the identities do not reduce to anything else. This enrichment is essential for capturing possibly nondeterministic computations, which preclude reducing the reduction preorder to a conversion relation. However, it may be that sharper specifying of the reduction preorder is needed. With its current axioms, it only precludes some models, but does not really change anything in the theory of abstraction: the statements simply go through enriched.

In principle, the main point of this theory is to *eliminating the abstraction as an imposed structure, for the sake of an intrinsic property.* In the action calculus, just like in the λ-calculus, the functional completeness is enforced by the abstraction operators; and just like there, these operators can be omitted from the presentation, and recovered as definable.

Proposition 3.3 *A (static) action calculus $\mathbb{A}[X;\mathsf{ab}]$ is isomorphic with the extension $\mathbb{A}[X;\Theta]$, where $\mathbb{A}$ is an action category, and Θ is the set of equations*

$$\omega \circ x = \mathrm{id}_\top, \tag{23}$$

$$\Delta \circ x = x \otimes x, \tag{24}$$

$$C_k(b_1, \ldots) \circ (x \otimes m) = C\left(b_1(x \otimes m_1), \ldots\right), \tag{25}$$

on all $x \in X$ and all $C \in \mathcal{K}$.

Proof. Θ is the smallest set of equations satisfying condition 2.7(ii). $\mathbb{A}[X;\Theta]$ is thus the free functionally complete extension of the action category $\mathbb{A}$ by X. The equivalent conditions of proposition 2.2 are thus satisfied for every $x \in X$. We show that this setting coincides with the structure of action calculus.

First of all, axiom σ trivially implies equation (5). The converse follows from 2.2(c).

Axioms γ and δ follow from the uniqueness part of 2.2(a). The converse uses the functoriality of ab_x.

Finally, axiom ζ corresponds to condition (15), while (14) remains implicit in the definition of a uniform ab_x in all contexts. □

Control structures [16]. While an action calculus is built up inductively, as the extension of a given $\mathcal{K}_\otimes$-category by names and abstractions, a (static) control structure is a $\mathcal{K}_\otimes$-category *readily given* with the abstraction functors and with names as distinguished arrows. The syntactic concepts, such as context, or substitution, are then reconstructed algebraically.

Corollary 3.4 *A $\mathcal{K}_\otimes$-category $\mathbb{C}$ is a (static) control structure if and only if it is equivalent with an extension $\mathbb{A}[X;Q]$ of a action category $\mathbb{A}$, with $Q \supseteq \Theta$ (23–25).*

Proof. By [16, thm. 4.15], $\mathbb{C}$ is a control structure if and only if it is a quotient of an action calculus $\mathbb{B}[X;\mathsf{ab}]$ along an abstraction preserving $\mathcal{K}_\otimes$-functor. Proposition 3.3 thus yields $\mathbb{C}$ as a quotient of $\mathbb{B}[X;\Theta]$. The subcategory $\mathbb{A}$, spanned in $\mathbb{C}$ by the image of $\mathbb{B} \longrightarrow \mathbb{B}[X;\Theta] \longrightarrow \mathbb{C}$ is an action category, since $\mathbb{B}$ is.

We show that $\mathbb{C}$ is a quotient of $\mathbb{A}[X]$. The functor $\mathbb{A}[X] \longrightarrow \mathbb{C}$ is induced as an extension of $\mathbb{A} \hookrightarrow \mathbb{C}$, by the universal property of polynomial categories. This functor is full and essentially surjective, because such a functor $\mathbb{B}[X] \longrightarrow \mathbb{C}$ factorises through it.

Letting Q identify two polynomials if and only if they are identified by $\mathbb{A}[X] \longrightarrow \mathbb{C}$, we get the equivalence $\mathbb{A}[X;Q] \longrightarrow \mathbb{C}$. □

3.2 Substitution

On the basis of the preceding results, the logical status of names can perhaps be determined a bit more precisely. While provided with the same abstraction power as variables, they are quite constrained as regards the substitution: only the renaming is allowed. In fact, a stronger rule than σ can be conservatively added.

Proposition 3.5 *Let $u : \ell \to k$ be a morphism in an action category $\mathbb{A}$, and x, y names of the types k and ℓ respectively. The substitution of $u \circ y$ for x induces a structure preserving functor $u^* : \mathbb{A}[x; \Theta] \to \mathbb{A}[y; \Theta]$ if and only if u is an admissible comonoid homomorphism.*

Therefore, $\mathbb{A}$ is a cartesian category if and only if all of its morphisms induce substitution functors.

Proof. Because of the functional completeness, the structure preserving functors $\mathbb{A}[x; Q] \longrightarrow \mathbb{A}[y; Q]$ correspond to the structure preserving functors between the Kleisli categories $\mathbb{A}_{k\otimes} \longrightarrow \mathbb{A}_{\ell\otimes}$. But such functors are in a bijective correspondence with the admissible comonoid homomorphisms $\ell \to k$. (This is routine category theory: the arrow part of lemma 2.4.) □

Restricted to the admissible comonoid homomorphisms, the action calculus thus supports the usual substitution mechanism: names behave like variables. Indeed, we have already seen that the comonoid homomorphisms in Rel are just the old functions. As an action category, Rel thus contains all of the world of substitution along functions, with the names playing the role of the variables. *But* it also contains much more.

The σ-rule can thus be consistently strengthened by allowing any comonoid homomorphism $u : \ell \to k$ for substitution. Conversely, any action $u : \ell \to k$, allowed for substitution, becomes a comonoid homomorphism. An action calculus with an unconstrained σ-rule thus degenerates into a cartesian category.

Allowing the substitution of arbitrary actions in the form $d : \top \to k$, as in [20, sec. 8], does not cause such degeneration immediately, but it does so in the presence of extensional higher order, as Milner poinst out in [*ibidem*, sec. 10]. However, the essence of the problem does not seem to lie in the extensionality, as suggested, but rather in the unconstrained substitution. Without either extensionality or higher order, the unconstrained substitution of data $d : \top \to k$ alone will lead to cartesianness as soon as there are enough such data to *generate*, i.e. ensure that

$$\forall d. u \circ d = v \circ d \quad \Longrightarrow \quad u = v.$$

And the unconstrained substitution of all $u : \ell \to k$ will of course enforce cartesiannes without any further assumptions, by proposition 3.5 alone. And cartesianness computationally means determinism. This can be formalised in terms of computational monads, but already the fact that the cartesian kernel of Rel consists of the functional relations provides an idea.

3.3 Examples

i. Since the comonoid structures in a cartesian category are unique, it is not just semi-cartesian, but also an action category — in a unique way, and assuming no controls. The action calculi, described in [18, 3.1–3.3], are of this kind: cartesian categories, or their polynomial extensions, supporting the κ-calculus. No controls.

ii. The departure from the cartesian setting is essentially due to introducing controls. In spite of their diversity, all the original action calculi can be subsumed under a simple syntactic construction, *molecular forms* [21] — which actually yields the free extension of a free cartesian category by a given set of controls. Such extensions turn out to be action categories, of course.

Consider a formal expression $(\vec{x})\langle\vec{y}\rangle$, where all x_i from $\vec{x} = x_0 \cdots x_{m-1}$ are distinct, whereas each y_j from of $\vec{y} = y_0 \cdots y_{n-1}$ occurs in $\vec{x}$ as some x_i. Setting $f(j) = i$ whenever $y_j = x_i$, we get a function $f : n \longrightarrow m$. The morphisms $m \longrightarrow n$ of the free strictly cartesian category $\mathbb{N}$, described in subsection 2.4, can thus be presented as expressions $(\vec{x})\langle\vec{y}\rangle$, modulo the renaming of $\vec{x}$, which is just the α-conversion, provided that $(\vec{x})$ is construed as a binding operator.

In fact, this is a rudimentary version of the κ-calculus, with $(\vec{x})$ as $\kappa\vec{x}$. The closed terms correspond to the arrows of $\mathbb{N}$, while a term $(\vec{x})\langle\vec{y},\vec{z}\rangle$, with $\vec{y} \subseteq \vec{x}$ and $\vec{z} \cap \vec{x} = \emptyset$, belongs to the polynomial extension $\mathbb{N}[\vec{z}]$. On the other hand, if the names from $\vec{x}, \vec{y}, \vec{z}$ are typed from some set Γ, the closed part of the calculus will yield the free strictly cartesian category over Γ. The rest of the calculus will give its polynomial extensions.

Milner's molecular forms are in principle obtained by adding suitable control expressions, the *molecules*, to the terms of this κ-calculus. For details, the reader is referred to [21]. All action calculi provided so far can be presented as calculi of molecular forms. On the other hand, closed molecular forms form action categories: the canonical comonoid structures $\omega = (x)\langle\rangle$ and $\Delta = (x)\langle xx\rangle$ are actually inherited from the free cartesian construction. The open forms form the algebraic extensions of these action categories by names.

As the calculus of molecular forms thus boils down to the κ-calculus plus controls, the action categories derived from it are actually not far from being the term models of action calculi.

iii. Moving to the less syntactical examples, we must first of all drop the strictness assumption: as everyone knows, monoidal categories coming about in nature tend to be non-strict. Nevertheless, the developed theory applies: in view of the coherence of monoidal categories, it is *a priori* clear that all stated results and definitions remain valid with the canonical isomorphisms introduced whenever needed. Alternatively, one could use the fact that every monoidal category is equivalent to a strict one[3] and extract strictly monoidal examples from the concrete ones.

[3] If a monoidal category $\mathbb{V}$ is viewed as a bicategory with one 0-cell, its strict version can be obtained as its image under the bicategorical Yoneda embedding. Roughly, an object (1-cell) $i \in \mathbb{V}$ is represented by the endofunctor $i \otimes (-) : \mathbb{V} \longrightarrow \mathbb{V}$, so that the tensor product becomes the endofunctor composition, which is, of course, strict.

The first and the most basic concrete example is the mentioned category Rel. Its monoidal structure and the commutative comonoids are induced by the cartesian structure of Set. But note that these inherited comonoids are not the only ones in Rel. In general, binary relations $\omega : k \nrightarrow 1$ and $\Delta : k \nrightarrow k \otimes k$ (i.e. $\omega \subseteq k$ and $\Delta \subseteq k \times k \times k$) will form a commutative comonoid if and only if for every $a \in k$ holds

$$\omega(a) \iff \forall bc. \Delta(a,b,c) \Leftrightarrow a = b = c \tag{26}$$

$$\neg\omega(a) \iff \exists! b.\ \omega(b) \wedge \Delta(a,a,b) \wedge \Delta(a,b,a). \tag{27}$$

These nonstandard comonoid structures offer a choice of nonstandard action category structures on Rel, with varying notions of name, extension etc. In fact, the morphisms from m to n in the extension of Rel by any name $x : 1 \nrightarrow k$ will always be the k-indexed families of relations $m \nrightarrow n$, but the composition and the identities will vary with the comonoid on k. The reader can work this out using 2.2(c), or noticing that the name x, as an arrow of the extension, is the family $\{x_a : 1 \nrightarrow k\}_{a \in k}$, where each $x_a \subseteq k$ is

$$x_a(b) \iff a = b \wedge \omega(a). \tag{28}$$

Which controls can be added to Rel? In principle, this is a bit like asking which operations can be added to the signature of a given algebra. For action categories there is a canonical choice, though. The details will be explained in [27], but let us here just display the control suitable for Rel (since this will perhaps suffice for some readers). The arity is

$$\frac{\varrho : m \nrightarrow \wp n}{U\varrho : m \nrightarrow n}$$

where $\wp n = \{n' \subseteq n\}$ is the power set. The idea is that the $U\varrho$-class of $a \in m$ should be the union of its ϱ-classes

$$U\varrho(a,b) \iff \exists n' \in \wp n.\ \varrho(a,n'). \tag{29}$$

Since it leaves the domain m of ϱ unchanged, U induces the family $\{U_k\}$, where each U_k is just U itself, restricted to the relations in the form $\varrho_k : k \otimes m \nrightarrow n$. Condition (18) is immeditate, and U is a control.

Mimicking the constructions of Rel over Set in more general settings yields more examples. In the rest of this section, we consider two such constructions, and the resulting action categories.

v. Abstract calculi of binary relations, presented as subobjects of products, can be developed in arbitrary toposes, varieties, regular categories in general [6]; even regular fibrations accomodate a similar construction [25]. One always gets an action category, with the standard comonoid structures inherited from the cartesian base, and with a choice of nonstandard comonoids, characterised by the formulas

$$\omega\Delta^o \quad = \quad \omega \otimes \omega \qquad (30)$$

$$\exists\varphi : k \mathrel{+\!\!\!\rightarrow} k \ \ \forall\overline{\omega} : k \mathrel{+\!\!\!\rightarrow} 1.$$

$$\Big(\varphi\varphi^o \subseteq \mathrm{id} \ \wedge \ \overline{\omega} \cap \omega = \emptyset \quad \Longrightarrow \quad \overline{\omega}\Delta^o \quad = \quad (\varphi\overline{\omega} \otimes \omega) \cup (\omega \otimes \varphi\overline{\omega})\Big) \quad (31)$$

which boil down to (26–27) when the classical logic is available.

In general, any allegory [6] or cartesian bicategory [2] subsumes an action category: the latter structure even contains commutative comonoids as a part of the definition. A range of familiar examples is obtained: categories of semilattices, total orders, total relations, partial maps... — they all turn out to support the name abstraction, with various classes of controls. The basic *interaction categories* [1] also fall into this group; SProc even appears as a category of relations for a certain regular fibration [26].

At this point, however, it is probably fair to reiterate that these considerations are restricted to the *static* action calculus. The abundance of the examples suggests that the presented theory of abstraction is actually *too* general to really pin down the intended computational meaning of the full action calculus. As pointed out before, an essential part remains to be captured by narrowing down the dynamics. It conceptual importance is can be seen, e.g., in the fact that the intuitive difference of interaction categories and action calculi can only be captured on the level of dynamics, since interaction categories support the static action calculus, but *fail to satisfy the dynamic axioms.* In fact, very few of the mentioned examples, with their natural hom-set orders, satisfy these axioms. Already in Rel, the identities are neither minimal nor maximal with respect to the inclusion. They are maximal among the partial maps, and minimal among the total relations, but more subtle dynamic requirements will perhaps be needed to really capture ideas.

vi. A different class of examples is obtained by generalising Rel as the Kleisli category for the commutative monad [9] $\wp$: Set $\to$ Set. The generalisation is based on the following lemma (roughly from [29, 4.7]).

Lemma 3.6 *A monad T on a monoidal category $\mathbb{V}$ is commutative if and only if the Kleisli category $\mathbb{V}_T$ and the canonical functor $\mathbb{V} \to \mathbb{V}_T$ are monoidal.*

Since a monoidal functor preserves comonoids, $\mathbb{V}_T$ will be an action category (semi-cartesian) as soon as $\mathbb{V}$ is.

According to Moggi [23], many notions of computation are naturally presented as strong monads on cartesian categories: the objects are sets of values, and the monad assigns to each of them the corresponding set of computations. If the base category depicts the maps on values, the induced Kleisli category can be thought of as the *category of computations.* The above lemma now implies that such a category of computations is an action category as soon as the corresponding monad is commutative. And the commutativity in this setting corresponds to the invariance of the order of execution.

This correspondence between commutative computational monads and action calculi is pursued in [27]: we show that there is a sense in which they are actually equivalent. It remains to be seen whether anything can be gained by extending Moggi's metalanguages, drawn from his monads, by features drawn from action calculi: name abstraction, perhaps controls. And whether the action calculus can be generalised from monoidal to *premonoidal* categories [29], corresponding to general, not necessarily commutative Moggi monads.

4 Related work

The "cartesian closed connection" for action calculi, has been pursued for some time, on type-theoretical [7, Concluding Remarks] or semantical [8, Introduction] grounds. The control calculi themselves belong to the latter line of research. We have shown that the structure of the action calculus can be analysed into (i) the closed part and (ii) its extensions by names — just like the λ-calculus can be decomposed into the cartesian closed structure and its polynomial extensions. As mentioned before, the essential ideas for capturing (i) are due to Philippa Gardner [7]: action categories are just the semantic counterpart of her closed action calculi. However, the notion of extension was beyond the scope of her work, and the actual reconstruction of the whole calculus from the closed part has not been considered.

Claudio Hermida and John Power [8] have tried to eliminate names in a different manner: not by abstracting them away and studying the closed part, but by presenting them categorically. Roughly, they represent the actions with a free name $x : \top \to k$ as the Kleisli arrows for the comonad $k \otimes (-)$. All such Kleisli categories are then arranged into an indexed category — a *fibrational* control structure. More recently [28], Power has proposed a simplification, *elementary* control structure, where these Kleisli categories are glued in one, rather than indexed. Either way, we get a reorganised copy of the original control structure, with the naming and the abstraction operators encoded in a reference-free fashion. Such encodings are sometimes very important, like e.g. de Bruijn indices, but they should not be confused with the actual elimination of the references, like in combinatory logic or categorical semantics. Power and Hermida's constructions do not fit in the standard semantical framework: applying them them to a control structure induced by a typed λ-calculus does not yield a cartesian closed category, but a categorical version of the whole calculus, with the Kleisli abstraction instead of the usual, reference-driven one. The underlying cartesian closed category is contained as a very small part: in the fibrational case, a single fibre. In general, the action category underlying a control structure is contained as a single fibre of its fibrational form. The point is that all of the structure is also contained in this small part, *in posse*, and can be recovered from it, with the abstraction derivable from its functional completeness.

References

[1] S. Abramsky et al., Interaction categories and the foundations of the typed concurrent programming, in: *Deductive Program Design: Proceedings of the 1994 Marktoberdorf International Summer School*, ed. M. Broy, (Springer 1995), (also on URL `http://theory.doc.ic.ac.uk`)

[2] A. Carboni and R.F.C. Walters, Cartesian bicategories I, *J. Pure Appl. Algebra* 49(1987) 11–32

[3] P.-L. Curien, *Combinateurs Catégoriques, Algorithmes Séquentiels et Programmation Applicative*, Thèse de Doctorat détat (Univ. Paris VII); updated English version: *Categorical Combinators, Sequential Algorithms and Functional Programming*, Progress in Theor. Comp. Sci. (Birkhäuser 1993)

[4] H.B. Curry, Grundlagen der Kombinatorischen Logik, *Amer. J. Math.* 52(1930) 509–536, 789–834

[5] H.B. Curry et al., *Combinatory Logic, Vol. I.*, Studies in Logic and Found. of Math. (North-Holland 1968)

[6] P.J. Freyd and A. Scedrov, *Categories, Allegories*, North-Holland Mathematical Library 39 (North-Holland, 1990)

[7] P. Gardner, A name-free account of action calculi, *Electronic Notes in Theor. Comp. Sci.* (URL `http://www.elsevier.nl:80/mcs/tcs/pc/`) 1(1995)

[8] C. Hermida and J. Power, Fibrational Control Structures, *CONCUR '95*, Lecture Notes in Computer Science 962 (Springer 1995) 117–129

[9] A. Kock, Monads on Symmetric Monoidal Closed categories, *Archiv der Mathematik* 21(1970) 1–10

[10] J. Lambek, Functional completeness of cartesian categories, *Ann. Math. Logic* 6(1974) 259–292

[11] J. Lambek, From types to sets, *Adv. in Math.* 36(1980) 113–164

[12] J. Lambek and B.A. Rattray, Functional completeness and Stone duality, *Studies in Foundations and Combinatorics. Advances in Math. Suppl. Studies* 1(1978) 1–9

[13] J. Lambek and P.J. Scott, *Introduction to higher order categorical logic*, Cambridge Stud. Adv. Math. 7 (Cambridge University Press 1986)

[14] F.W. Lawvere, *Functorial Semantics of Algebraic Theories*, Thesis (Columbia University 1963)

[15] S. Mac Lane. *Categories for the Working Mathematician*, Graduate Texts in Mathematics 5 (Springer 1971)

[16] A. Mifsud et. al. Control Structures, *Proceedings of LICS '95* (IEEE 1995)

[17] R. Milner, *Communication and Concurrency* (Prentice Hall 1989)

[18] R. Milner, Action Calculi and the π-Calculus, in: *Proceedings of the NATO Summer School on Logic and Computation*, (Springer 1993)

[19] R. Milner, Action Calculi I: Axioms and Applications, *appeared as:* Action calculi, or syntactic action structures, *Proceedings of MFCS '93*, Lecture Notes in Computer Science 711 (Springer 1993) 105–121

[20] R. Milner, Action Calculi III (URL `http://theory.doc.ic.ac.uk`)

[21] R. Milner, Action Calculi IV (URL `http://theory.doc.ic.ac.uk`)

[22] R. Milner et al., A calculus of mobile processes I and II, *Inform. and Comput.* 100(1992), 1–77

[23] E. Moggi, Notions of computation as monads, *Inform. and Comput.* 93(1991) 55–92

[24] J. Meseguer and U. Montanari, Petri Nets are Monoids, *Inform. and Comput.* 88/2(1990) 105-155

[25] D. Pavlović, Maps II: Chasing diagrams in categorical proof theory, *J. of the IGPL* 4/2(1996), 1–36 (URL `http://www.mpi-sb.mpg.de/igpl/Journal`)

[26] D. Pavlović, Categorical logic of concurrency and interaction I. Synchronous processes, in: *Theory and Formal Methods of Computing 1994*, C.L. Hankin et al., eds. (World Scientific 1995), 105–141

[27] D. Pavlović, Action calculi and computational monads, *in preparation*

[28] J. Power, Elementary Control Structures, *to appear* in the proceedings of CONCUR '96

[29] J. Power and E. Robinson, Premonoidal categories, *manuscript*

[30] V. Pratt, Chu Spaces and their Interpretation as Concurrent Objects, *in:* J. van Leeuwen (ed.), *Computer Science Today: Recent Trends and Developments*, Lecture Notes in Computer Science 1000 (Springer 1995) 392–405

[31] M. Schönfinkel, Über die Bausteine der Mathematischen Logik, *Math. Annalen* 92(1924) 305–316

[32] D. Scott, Relating Theories of the λ-Calculus, in: *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, eds. J.P. Seldin and J.R. Hindley (Academic Press 1980) 403–450