States and exceptions are dual effects

Jean-Guillaume Dumas, Dominique Duval, Laurent Fousse, Jean-Claude Reynaud

States and exceptions are dual effects

Jean-Guillaume Dumas, Dominique Duval, Laurent Fousse, Jean-Claude Reynaud

LJK, University of Grenoble, France

January 11., 2010

Abstract. Global states and exceptions form two basic computational effects. In this paper it is proved that they can be seen as dual to each other: the lookup and update operations for global states are dual to the raise and handle operations for exceptions, respectively. In order to get this result we use a monad for exceptions and a comonad for global states.

1 Introduction

The denotational semantics of languages with *computational effects* can be expressed in categorical terms, thanks to *monads* [7]: for instance, global states correspond to the monad on **Set** with endofuntor $T(X) = (X \times St)^{St}$ where St is the set of states, while exceptions correspond to the monad on **Set** with endofuntor $T(X) = X + Exc$ where Exc is the set of exceptions. Each computational effect comes with its associated operations: for instance, the operations lookup and update for the states, the operations raise and handle for the exceptions. Another approach is based on *Lawvere theories*; this point of view is related to monads by an adjunction [6]. However, there are still several issues for designing a categorical semantics for computational effects. One of these issues is that some of the operations associated with the effects are not *algebraic*, as defined in [9]: for instance, lookup, update and raise are algebraic operations, while handle is not algebraic [8, 9].

A computational effect relies on a kind of translation from a situation where the effect is partially hidden to a situation where it becomes explicit: for instance, there is no explicit type S for states in an imperative language, however the set of states St does appear explicitly in the semantics for global states. In this paper, we focus on two computational effects: global states and exceptions. First, in section 2, these two effects are treated in an explicit way. We prove that they can be seen as mutually *dual*, in the sense that their semantics is defined by the models of two dual algebraic specifications. More precisely, we use *sketches* as algebraic specifications, since this provides a clean treatement of sums as dual to products. The idea is that $X \times St$ for a fixed St is dual to $X + Exc$ for a fixed Exc, and that this duality can be extended to the operations: lookup is dual to raise, and update is dual to handle. Then, in section 3, we come back to the proper effects, with hidden features. For this purpose, as in the classical approach we use the monad $T(X) = X + Exc$ for the exceptions, but we use the dual *comonad* $T(X) = X \times St$ for the global states. We prove that the duality can be easily expressed from this point of view.

To our knowledge, the fact that global states and exceptions are dual computational effects is a new result. It has been necessary to use both monads and comonads for getting this result in its right setting, i.e., when the effects are hidden. We would like to suggest that, while monads are indeed one major tool for expliciting effects, additional tools like comonads or other categorical features may also be helpful.

2 Duality

In this section the effects are not seen as such, since they are explicit in the specifications. We use sketches [1] as specifications: a sketch with finite products **S** for global states and dually a sketch with finite sums **E** for exceptions. The category of (set-valued) models of a sketch **Sp** is denoted Mod(**Sp**). When $\mathbf{Sp}_0$ is a subsketch of **Sp** and M_0 a model of $\mathbf{Sp}_0$, we denote $\mathrm{Mod}(\mathbf{Sp})|_{M_0}$ the subcategory of Mod(**Sp**) made of the models M of **Sp** which coincide with M_0 on $\mathbf{Sp}_0$ and of the morphisms which extend the identity of M_0. We often write $M\ldots$ for $M(\ldots)$ when M is a model of a sketch. In order to focus on the usual semantics of effects in their simplest setting, we look only at set-valued models. However they could easily be replaced by the models in a category **C** with the required products (for global sates) and sums (for exceptions).

2.1 Global states

Let $Loc = \{i\}$ be a set, called the set of *locations*.

Definition 2.1. Let $\mathbf{S}_0$ denote the specification simply made of:
– a point V_i for each $i \in Loc$, called the type of *values* of i.
The *specification for global states* $\mathbf{S}$ is made of $\mathbf{S}_0$ and:
– a point S, called the type of *states*,
– an arrow $l_i : S \to V_i$ for each $i \in Loc$, called the *lookup* at i,
– an arrow $u_i : V_i \times S \to S$ for each $i \in Loc$, called the *update* at i,
– an equation $l_i \circ u_i = pr_{V_i}$ for each $i \in Loc$,
– an equation $l_j \circ u_i = l_j \circ pr_S$ for each pair $(i, j) \in Loc^2$ with $i \neq j$,
where $pr_{V_i} : V_i \times S \to V_i$ and $pr_S : V_i \times S \to S$ are the projections.

Remark 2.2. A model M_0 of $\mathbf{S}_0$ is simply made of a set $Val_i = M_0V_i$ for each $i \in Loc$, while a model of $\mathbf{S}$ is made of a set $Val_i = MV_i$ for each $i \in Loc$, a set $St = MS$, and for each $i \in Loc$ two functions $Ml_i : St \to Val_i$ and $Mu_i : Val_i \times St \to St$ such that:
– $Ml_i(Mu_i(x_i, s)) = x_i$ for each $x_i \in Val_i$ and $s \in St$
– $Ml_j(Mu_i(x_j, s)) = Ml_j(s)$ for each $j \in Loc$ such that $j \neq i$, each $x_j \in Val_j$ and $s \in St$.

Proposition 2.3. *Let M_0 be a model of $\mathbf{S}_0$, made of a set $Val_i = M_0V_i$ for each $i \in Loc$. The category $\mathrm{Mod}(\mathbf{S})|_{M_0}$ has a terminal object denoted $[[\,]]$, such that:*
– $[[S]] = \prod_j Val_j$,
– $[[l_i]] : \prod_j Val_j \to Val_i$ *is the projection, for each* $i \in Loc$,
– $[[u_i]] : Val_i \times \prod_j Val_j \to \prod_j Val_j$ *maps* $(x_i, (x_j)_j)$ *to* $(z_j)_j$ *where* $z_i = x_i$ *and* $z_j = y_j$ *for every* $j \neq i$, *for each* $i \in Loc$.

It follows from remark 2.2 and proposition 2.3 that the next definition corresponds to the usual semantics for global states in algebraic specifications.

Definition 2.4. Let M_0 be a model of $\mathbf{S}_0$. The category of *loose semantics for global states above* M_0 is the category $\mathrm{Mod}(\mathbf{S})|_{M_0}$ of models of $\mathbf{S}$ above M_0. The *terminal semantics for global states above* M_0 is the terminal model of $\mathbf{S}$ above M_0.

2.2 Exceptions

Now the *specification for exceptions* is defined as the *dual* of the specification for global states. For readability, the names of the points and arrows are changed.
Let $Etype = \{i\}$ be a set, called the set of *exception types*.

Definition 2.5. Let $\mathbf{E}_0$ denote the specification simply made of:
– a point P_i for each $i \in Etype$, called the type of *parameters* for exceptions of type i.
The *specification for exceptions* $\mathbf{E}$ is made of $\mathbf{E}_0$ and:
– a point E, called the type of *exceptions*,
– an arrow $r_i : P_i \to E$ for each $i \in Etype$, called the *raising* of exceptions of type i,
– an arrow $h_i : E \to P_i + E$ for each $i \in Etype$, called the *handling* of exceptions of type i,
– an equation $h_i \circ r_i = in_{P_i}$ for each $i \in Etype$,
– an equation $h_i \circ r_j = in_E \circ r_j$ for each pair $(i, j) \in Etype^2$ with $i \neq j$,
where $in_{P_i} : P_i \to P_i + E$ and $in_E : E \to P_i + E$ are the coprojections.

Remark 2.6. A model M_0 of $\mathbf{E}_0$ is simply made of a set $Par_i = M_0P_i$ for each $i \in Etype$, while a model of $\mathbf{E}$ is made of a set $Par_i = MP_i$ for each $i \in Etype$, a set $Exc = ME$, and for each $i \in Etype$ two functions $Mr_i : Par_i \to Exc$ and $Mh_i : Exc \to Par_i + Exc$ such that (writing explicitly the inclusions as $in_{Par} : Par_i \to Par_i + Exc$ and $in_{Exc} : Exc \to Par_i + Exc$ in order to avoid ambiguity):

– $Mh_i(Mr_i(x_i)) = in_{Par_i}(x_i)$ for each $x_i \in Par_i$
– $Mh_i(Mr_j(x_j)) = in_{Exc}(Mr_j(x_j))$ for each $j \in Etype$ such that $j \neq i$ and each $x_j \in P_j$.
This means that the handling function Mh_i runs as follows. For each $e \in Exc$, the result of $Mh_i(e)$ identifies whether e is an exception of type i, if so then $Mh_i(e)$ returns the parameter $x_i \in Par_i$ such that $e = Mr_i(x_i)$, otherwise $Mh_i(e)$ returns $e \in Exc$ without analyzing it any further.

Proposition 2.7. *Let M_0 be a model of $\mathbf{E}_0$, made of a set $Par_i = M_0P_i$ for each $i \in Etype$. The category $\mathrm{Mod}(\mathbf{E})|_{M_0}$ has an initial object denoted $[[\,]]$, such that:*
– $[[E]] = \sum_j Par_j$,
– $[[r_i]] : Par_i \to \sum_j Par_j$ *is the coprojection, for each $i \in Etype$,*
– $[[h_i]] : \sum_j Par_j \to Par_i + \sum_j Par_j$ *maps $x_i \in Par_i$ to $x_i \in Par_i$ (in the first summand) and $x_j \in \sum_j Par_j$ to $x_ij \in \sum_j Par_j$ (in the second summand) when $j \neq i$, for each $i \in Etype$.*

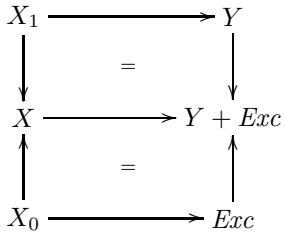
We claim that the next definition corresponds to the usual semantics of exceptions in programming languages. This claim is supported by proposition 2.9.

Definition 2.8. Let M_0 be a model of $\mathbf{E}_0$. The category of *loose semantics for exceptions above* M_0 is the category $\mathrm{Mod}(\mathbf{E})|_{M_0}$ of models of $\mathbf{E}$ above M_0. The *initial semantics for exceptions above* M_0 is the initial model of $\mathbf{E}$ above M_0.

For a while, let us forget about the previous sections and come back to the usual meaning of exceptions, in an explicit set-valued context. Let Exc be a set, called the *set of exceptions*, with a function $r_i : Par_i \to Exc$ for *raising an exception of type i*. Let $f : X \to Y + Exc$ be some function; for each $x \in X$, if $f(x) = e \in Exc$ then we say that $f(x)$ *raises the exception e*. Let $i \in Etype$, and let $g : X \times Par_i \to Y + Exc$, then g may be used to handle an exception raised by f if this exception is of type i; it should be noted that g itself may raise an exception. The fact of using g for handling an exception of type i raised by f means that instead of $f : X \to Y + E$ we call a function, which will be denoted f `handle` $[i \Rightarrow g] : X \to Y + E$, defined as follows. For each $x \in X$:
– if $f(x)$ does not raise any exception then $(f$ `handle` $[i \Rightarrow g])(x) = f(x) \in Y$,
– otherwise if $f(x)$ raises an exception of the form $r_i(x_i)$ for some $x_i \in Par_i$, then $(f$ `handle` $[i \Rightarrow g])(x) = g(x, x_i) \in Y + Exc$,
– otherwise (i.e., if $f(x)$ raises an exception e which is not of type i), then $(f$ `handle` $[i \Rightarrow g])(x) = f(x) \in Exc$.
The handling of several types of exceptions is easily obtained by iterating the construction f `handle` $[i \Rightarrow g]$.

Proposition 2.9. *Let us consider a model M of the specification $\mathbf{E}$, and let $Par_i = MP_i$ for each i and $Exc = ME$. Let us consider two sets X, Y and two functions $f : X \to Y + Exc$ and $g : X \times Par_i \to Y + Exc$. Then f `handle` $[i \Rightarrow g]$, as define above, can be built from M, f and g, using only the fact that* **Set** *has sums of the form $\ldots + Exc$ and that these sums are well-behaved.*

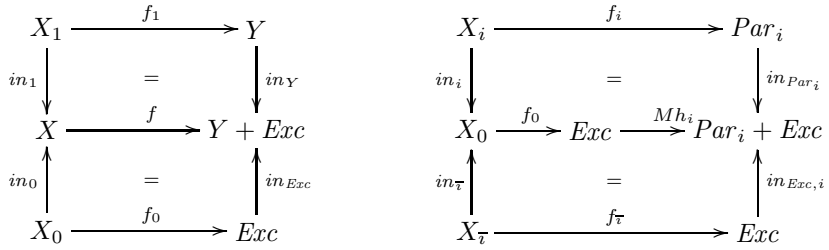
The sums $\ldots + Exc$ in $\mathbf{C}$ are *well-behaved*, in the sense of extensive categories [2], if for each commutative diagram:



where the right column is the sum, the two squares are pullbacks if and only if the left column is a sum.

Proof. First, the set X is decomposed as $X = X_1 + X_i + X_{\overline{i}}$ thanks to the well-behaved property of sums,

applied twice. Note that Mh_i is used in the second diagram.



Then, a function $H : X \to Y + Exc$ is defined as $H = [H_1|H_i|H_{\overline{i}}] : X_1 + X_i + X_{\overline{i}} \to Y + Exc$:
— $H_1 : X_1 \to Y + Exc$ is $H_1 = f \circ in_1 = in_Y \circ f_1$, which means that $H(x) = f(x) \in in_Y(Y)$ when f does not raise any exception,
— $H_i : X_i \to Y + Exc$ is $H_i = g \circ (in_0 \circ in_i, f_i)$, which means that $H(x) = g(x, x_i) \in Y + Exc$ when $f(x)$ raises an exception $r_i(x_i)$ of type i,
— $H_{\overline{i}} : X_{\overline{i}} \to Y + Exc$ is $H_{\overline{i}} = in_{Exc} \circ f_{\overline{i}}$, which means that $H(x) = f(x) \in in_{Exc}(Exc)$ when $f(x)$ raises an exception which is not of type i,. □

2.3 Duality

Theorem 2.10. *The semantics of global states and the semantics of exceptions are dual.*

Proof. Definition 2.4 provides the semantics of global states, definition 2.8 is its dual, and proposition 2.9 shows that it does provide the semantics of exceptions. This yields the result for the loose semantics, then the result follows for the specific semantics since terminal is dual to initial. □

3 Effects

In this section we define global states and exceptions as effects, with hidden types S and E, respectively. We show that their semantics, as defined in section 2, can also be defined directly from this point of view, so that the duality theorem 2.10 actually is a theorem about effects. It can be assumed that the base category $\mathbf{C}$ is $\mathbf{Set}$.

3.1 Decorated categories

Given a monad T on a category $\mathbf{C}$, the canonical functor from $\mathbf{C}$ to the Kleisli category $\mathbf{C}_T$ of T is the identity on objects. Following [7], the morphisms of $\mathbf{C}_T$ may be called the *computations* and the morphisms in the image of $\mathbf{C}$ the *values* (so that each value is a computation). The values are sometimes also called the *pure* morphisms. This classification of morphisms is now generalized (for a more subtle use of the notion of decoration, see [3]).

Definition 3.1. In this paper, a *decorated category* $\mathbf{C}^{dec}$ is made of three nested categories with the same objects $\mathbf{C}^p \subseteqq \mathbf{C}^q \subseteqq \mathbf{C}^r$ (we use "$\subseteqq$" for an inclusion which is the identity on objects). A morphism in $\mathbf{C}^d$, for every $d \in \{p, q, r\}$, is denoted f^d, and the symbol d is called a *decoration* of f^d. Clearly every f^p is also an f^q and every f^q is also an f^r, and the identities are in $\mathbf{C}^p$. A *decorated specification* $\mathbf{Sp}^{dec}$ is a sketch where each arrow has at least one decoration $d \in \{p, q, r\}$.

Let $\mathbf{C}$ be a category and (T, η, μ) a monad on $\mathbf{C}$. Then in $\mathbf{C}$, every morphism $f_p : X \to Y$ gives rise to $\eta_Y \circ f_p : X \to TY$, every morphism $f_q : X \to TY$ gives rise to $\mu_Y \circ Tf_q : TX \to TY$, and when $f_q = \eta_Y \circ f_p$ then $\mu_Y \circ Tf_q = \mu_Y \circ T\eta_Y \circ Tf_p = Tf_p$. This yields a decorated category $D_T(\mathbf{C})$, as defined below, where essentially $\mathbf{C}^p$ is $\mathbf{C}$ and $\mathbf{C}^q$ is the Kleisli category of T.

Definition 3.2. Let $\mathbf{C}$ be a category and (T, η, μ) (simply denoted T) a monad on $\mathbf{C}$. Then $D_T(\mathbf{C}) = \mathbf{C}_T^p \subseteqq \mathbf{C}_T^q \subseteqq \mathbf{C}_T^r$ is the decorated category with the same objects as $\mathbf{C}$ such that:
– there is a morphism $f^r : X \to Y$ in $\mathbf{C}_T^r$ for each $f_r : TX \to TY$ in $\mathbf{C}$,
– such a morphism $f^r : X \to Y$ is in $\mathbf{C}_T^q$ if and only if $f_r = \mu_Y \circ Tf_q$ for some $f_q : X \to TY$ in $\mathbf{C}$,
– and such a morphism $f^q : X \to Y$ is in $\mathbf{C}_T^p$ if and only if $f_q = \eta_Y \circ f_p$ for some $f_p : X \to Y$ in $\mathbf{C}$.
The composition on $\mathbf{C}_T^r$ is the composition in $\mathbf{C}$, so that in $\mathbf{C}_T^p$ the composition is also as in $\mathbf{C}$, and in $\mathbf{C}_T^q$ it is the Keisli composition. In addition, the composition of a morphism in $\mathbf{C}_T^q$ with a morphism in $\mathbf{C}_T^r$ is in $\mathbf{C}_T^q$: $g^r \circ f^q = (g \circ f)^q$.

If there are enough sums in $\mathbf{C}$, then every family of morphisms $f_i^q : X_i \to Y$ gives rise to $[f_i]_i^q : \sum_i X_i \to Y$ .. characterized (up to isomorphism) by $[f_i]_i^q \circ in_i^p = f_i^q$ for each i, where $in_i^p : X_i \to \sum_i X_i$ is the injection.
Dually, each comonad (T, ε, δ) (or simply T) on a category $\mathbf{C}$ gives rise to a decorated category, still denoted $D_T(\mathbf{C})$.

3.2 Global states

Let $Loc = \{i\}$ be a set, called the set of *locations*.
Let $\mathbf{C}$ be a category with a terminal object 1, with a distinguished object St called the type of *states*, and a product-with-St functor $T(\ldots) = \ldots \times St$. Then T is the endofunctor of a comonad (T, ε, δ) where $\varepsilon_X : X \times St \to X$ is the projection and $\delta_X : X \times St \to X \times St \times St$ duplicates the St-component. So, we get a decorated category $D_{\ldots \times St}(\mathbf{C})$ as in section 3.1.

Definition 3.3. Let $\mathbf{Sp}^{dec}$ be a decorated specification. The *expansion* of $\mathbf{Sp}^{dec}$ for global states is the specification $E_{\ldots \times S}(\mathbf{Sp}^{dec})$ with the same points as $\mathbf{Sp}^{dec}$ and with:
– a point S,
– an arrow $f_p : X \to Y$ for each $f^p : X \to Y$ in $\mathbf{Sp}^{dec}$,
– an arrow $f_q : X \to Y \times S$ for each $f^q : X \to Y$ in $\mathbf{Sp}^{dec}$,
– an arrow $f_r : X \times S \to Y \times S$ for each $f^r : X \to Y$ in $\mathbf{Sp}^{dec}$,
– and similarly for the equations.

The following result is easy to check directly, it can also be obtained from an adjunction [4]. A decorated model of a decorated specification $\mathbf{Sp}^{dec}$ in a decorated category $\mathbf{C}^{dec}$ is defined like a model of a specification in a category which in addition preserves the decorations. This gives rise to the category of $\mathrm{Mod}^{dec}(\mathbf{Sp}^{dec}, \mathbf{C}^{dec})$.

Proposition 3.4. *Let $\mathbf{Sp}^{dec}$ be a decorated specification and $\mathbf{C}$ a category with a terminal object 1, a distinguished object St and a comonad $T(\ldots) = \ldots \times St$. Then there is a bijection:*

$$\mathrm{Mod}^{dec}(\mathbf{Sp}^{dec}, D_{\ldots \times St}(\mathbf{C})) \cong \mathrm{Mod}_{S \mapsto St}(E_{\ldots \times S}(\mathbf{Sp}^{dec}), \mathbf{C})$$

where $\mathrm{Mod}_{S \mapsto St}(E_{\ldots \times S}(\mathbf{Sp}^{dec}), \mathbf{C})$ is the full subcategory of $\mathrm{Mod}(E_{\ldots \times S}(\mathbf{Sp}^{dec}), \mathbf{C})$ made of the models which map S to St.

Definition 3.5. Let $\mathbf{S}_0^{dec}$ denote the decorated specification simply made of:
– a point V_i for each $i \in Loc$, called the type of *values* of i.
The *decorated specification for global states* $\mathbf{S}^{dec}$ is made of $\mathbf{S}_0^{dec}$ and:
– an arrow $l_i^q : 1 \to V_i$ for each $i \in Loc$, called the *lookup* at i,
– an arrow $u_i^r : V_i \to 1$ for each $i \in Loc$, called the *update* at i,
– an equation $l_i^q \circ u_i^r = id_{V_i}^p$ for each $i \in Loc$,
– an equation $l_j^q \circ u_i^r = l_j^q \circ (\,)_{V_i}^p$ for each pair $(i, j) \in Loc^2$ with $i \neq j$, where $(\,)_{V_i}^p : V_i \to 1$.

Then the expansion $E_{\ldots \times S}(\mathbf{S}^{dec})$ is the specification $\mathbf{S}$ from definition 2.1. So, proposition 3.4 state that $\mathrm{Mod}^{dec}(\mathbf{S}^{dec}, D_{\ldots \times St}(\mathbf{Set})) \cong \mathrm{Mod}_{S \mapsto St}(\mathbf{S}, \mathbf{Set})$ The next reuslt follows.

Corollary 3.6. *Let M_0 be a model of $\mathbf{S}_0^{dec}$. The category of loose semantics for global states above M_0 is the category* $\mathrm{Mod}^{dec}(\mathbf{S}^{dec}, D_{\ldots\times St}(\mathbf{Set}))|_{M_0}$ *of decorated models of $\mathbf{S}^{dec}$ above M_0.*

Remark 3.7. Usually the global state effect is formalized using the monad with endofunctor $T'(X) = (X \times S)^S$, assuming that there are exponentials of the form $(\ldots \times S)^S$ in $\mathbf{C}$. Up to currifying the Kleisli category of the monad T' can be identified to the category $\mathbf{C}^r$. So, with the point of view of monads, we get the inclusion $\mathbf{C}^p \subsetneqq \mathbf{C}^r$, but the intermediate category $\mathbf{C}^q$ has to be added.

3.3 Exceptions

Following the same lines as in section 2.2, the treatment of exceptions as effects is dual to the treatment of global states as effects in section 3.2,

Remark 3.8. It is usual to formalize the exceptions thanks to the monad $\ldots + E$. Usually the raising of exceptions is defined by operations $\mathtt{raise}_{i,X} : P_i \to X$ for each i and for each object X. This point of view is easily recovered from our approach, by defining $\mathtt{raise}_{i,X} = [\,]_X \circ r_i : P_i \to X$. But the handling of exceptions does not fit into the usual treatment of effects by monads because it is not an *algebraic operation* in the sense of [9]. On the other hand, [5] contains a preliminary version of the treatment of exceptions as in this paper.

References

[1] Michael Barr, Charles Wells. *Category Theory for Computing Science.* 3rd ed., Publications CRM PM023 (1999).

[2] A. Carboni, S. Lack, R.F.C. Walters. Introduction to extensive and distributive categories. *Journal of Pure and Applied Algebra* 84: 145-158 (1993).

[3] Jean-Guillaume Dumas, Dominique Duval, Jean-Claude Reynaud. Cartesian effect categories are Freyd-categories. *CoRR abs/0903.3311* (2009)

[4] Dominique Duval. Diagrammatic Specifications. *Mathematical Structures in Computer Science* (13): 857-890 (2003).

[5] Dominique Duval, Jean-Claude Reynaud. Diagrammatic logic and effects : the example of exceptions. Rapport de Recherche (21 dcembre 2004) ccsd-00004129.

[6] Martin Hyland, John Power. The Category Theoretic Understanding of Universal Algebra: Lawvere Theories and Monads. *Electr. Notes Theor. Comput. Sci.* 172: 437-458 (2007).

[7] Eugenio Moggi. Notions of Computation and Monads. *Inf. Comput.* 93(1): 55-92 (1991).

[8] Gordon D. Plotkin, John Power. Notions of Computation Determine Monads. FoSSaCS'02. *Lecture Notes in Computer Science* 2303: 342-356 (2002).

[9] Gordon D. Plotkin, John Power. Algebraic Operations and Generic Effects. *Applied Categorical Structures* 11(1): 69-94 (2003).

A Table

This table summarizes most notations used in the paper. When the two main columns are subdivided, the left hand-side is decorated while the right hand-side is explicit.

<table>
<tr><th colspan="2">Global states</th><th colspan="2">Exceptions</th></tr>
<tr><td colspan="4">Category</td></tr>
<tr><td colspan="2">$\mathbf{C}$ with 1, with S and $\ldots \times S$</td><td colspan="2">$\mathbf{C}$ with 0, with E and $\ldots + E$</td></tr>
<tr><td colspan="4">(Co)Monad</td></tr>
<tr><td colspan="2">$\mathbf{CoMonad}\ T(X) = X \times S$
$\varepsilon_X : X \times S \to X$, $\delta_X : X \times S \to X \times S \times S$</td><td colspan="2">$\mathbf{Monad}\ T(X) = X + E$
$\eta_X : X \to X + E$, $\mu_X : X + E + E \to X + E$</td></tr>
<tr><td colspan="4">$p \Rightarrow q \Rightarrow r$</td></tr>
<tr><td colspan="2">$(p)\quad X \to Y$
$(q)\quad X \times S \to Y$
$(r)\quad X \times S \to Y \times S$</td><td colspan="2">$(p)\quad X \to Y$
$(q)\quad X \to Y + E$
$(r)\quad X + E \to Y + E$</td></tr>
<tr><td colspan="4">“q”</td></tr>
<tr><td colspan="2">$\mathbf{lookup}$</td><td colspan="2">$\mathbf{raise}$</td></tr>
<tr><td>$l_i^q : 1 \to V_i$
$l^q = (l_i^q)_i : 1 \to \prod_i V_i$</td><td>$l_i : S \to V_i$
$l = (l_i)_i : S \to \prod_i V_i$</td><td>$r_i^q : P_i \to 0$
$r^q = [r_i^q]_i : \sum_i P_i \to 0$</td><td>$r_i : P_i \to E$
$r = [r_i]_i : \sum_i P_i \to E$</td></tr>
<tr><td colspan="4">“r”</td></tr>
<tr><td colspan="2">$\mathbf{update}$</td><td colspan="2">$\mathbf{handle}$</td></tr>
<tr><td>$u_i^r : V_i \to 1$
$\varphi_{i,j}^q : V_i \to V_j$
$\varphi_{i,i}^q = id_{V_i}^p$
$\varphi_{i,j\neq i}^q = l_j^q \circ (\,)_{V_i}$
$\varphi_i^q = (\varphi_{i,j}^q)_j$
$\begin{array}{ccc} V_i & \xrightarrow{\varphi_i^q} & \prod_j V_j \\ {\scriptstyle u_i^r}\downarrow & = & \downarrow{\scriptstyle id^p} \\ 1 & \xrightarrow{l^q} & \prod_j V_j \end{array}$</td><td>$u_i : V_i \times S \to S$
$\varphi_{i,j} : V_i \times S \to V_j$
$\varphi_{i,i} = pr_{V_i}$
$\varphi_{i,j\neq i} = l_j \circ pr_S$
$\varphi_i = (\varphi_{i,j})_j$
$\begin{array}{ccc} V_i \times S & \xrightarrow{\varphi_i} & \prod_j V_j \\ {\scriptstyle u_i}\downarrow & = & \downarrow{\scriptstyle id} \\ S & \xrightarrow{l} & \prod_j V_j \end{array}$</td><td>$h_i^r : 0 \to P_i$
$\psi_{i,j}^q : P_j \to P_i$
$\psi_{i,i}^q = id_{P_i}^p$
$\psi_{i,j\neq i}^q = [\,]_E^p \circ r_j^q$
$\psi_i^q = [\psi_{i,j}^q]_j$
$\begin{array}{ccc} \sum_j P_j & \xrightarrow{\psi_i^q} & P_i \\ {\scriptstyle id^p}\uparrow & = & \uparrow{\scriptstyle h_i^r} \\ \sum_j P_j & \xrightarrow{e^q} & 0 \end{array}$</td><td>$h_i : E \to P_i + E$
$\psi_{i,j} : P_j \to P_i + E$
$\psi_{i,i} = in_{P_i}$
$\psi_{i,j\neq i} = in_E \circ r_j$
$\psi_i = [\psi_{i,j}]_j$
$\begin{array}{ccc} \sum_j P_j & \xrightarrow{\psi_i} & P_i + E \\ {\scriptstyle id}\uparrow & = & \uparrow{\scriptstyle h_i} \\ \sum_j P_j & \xrightarrow{e} & E \end{array}$</td></tr>
<tr><td colspan="4">Remarks</td></tr>
<tr><td colspan="2">if $\forall i\ V_i = Val$
then $l : S \to Val^{Loc}$ (where $Loc = \{i\}$)</td><td colspan="2">if $\forall i\ P_i = Par$
then $e : Etype \times Par \to E$ (where $Etype = \{i\}$)</td></tr>
<tr><td colspan="2">if $l = (l_i)_i : S \to \prod_i V_i$ is terminal
then $l : S \xrightarrow{\simeq} \prod_i V_i$
and by coinduction: existence and unicity of u</td><td colspan="2">if $r = [r_i]_i : \sum_i P_i \to E$ is initial
then $e : \sum_i P_i \xrightarrow{\simeq} E$
and by induction: existence and unicity of h</td></tr>
</table>