Towards an Override in Topoi

Arthur Hughes

University of Dublin,

Trinity College, Dublin, Ireland

e-mail : Arthur.Hughes@cs.tcd.ie

November 5, 1998

Abstract

Models of software systems are built in Z and VDM using partial functions between sets and certain operations on these partial functions : extension ($\sqcup$), restriction ($\lhd$), removal ($\lhd$) and override ($\dagger$). Can these operations be given a categorial semantics? Doing so will show the ‘nature’ of the operations. The operation of override is found to depend on the ‘shape’ on X , the poset $\mathcal{P}X$. The operations are developed in an elementary topos $\mathcal{E}$. This is achieved by constructing each operation in the topos **Set**, of sets and total functions, and then using these constructions as the definition of the operations in an elementary topos. Each of the operations is thus given a categorial semantics. As

an example the operation of override is considered in the topos $\mathbf{Set}^{\downarrow}$, of total functions and commuting diagrams. Can models of software systems be built in topoi other than **Set**?

1 Prologue

The VDM (Jones 1990) and Z (Spivey 1992) notations have been used widely in the specification and development of software systems. These methods share with the *Irish School of Constructive Mathematics* ($\mathbf{M}_C^\clubsuit$), of which the *Irish School of the VDM* ($\mathbf{VDM}^\clubsuit$) (Mac an Airchinnigh 1990; Mac an Airchinnigh 1991; Mac an Airchinnigh 1993) is a part, a collection of mathematical operations: extend, restriction, removal and override. These operations are fundamental for modelling dynamically varying systems (Hughes and Donnelly 1998; Hughes and Pahl 1997; Poppleton 1997). We would like to give a categorial semantics to these operations.

Let us start by considering the operation of ‘overriding’ or *overwriting* a total function $f: X \rightarrow Y$ between two sets X and Y by the partial function $[x \mapsto y]$ where $x \in X$ and $y \in Y$, the resulting function $f \dagger [x \mapsto y]$ will have the existing value of x in the function f replaced by the ‘new’ value y . Historically, the override operator was introduced to give meaning to the assignment statement of programming languages.

Can we express the overridden function in terms of composition alone? In other words does the determination problem or choice problem (Lawvere and Schanuel 1997, page 45) have a solution? Consider first the determination problem

This determination problem will only have a solution if

$$|f^{-1}(\{f(x)\})| = 1.$$

When this is true $g = 1_Y \dagger [f(x) \mapsto y]$, where 1_Y denotes the identity function on the set Y . Now consider the choice problem

This choice problem will only have a solution if

$$f^{-1}(\{y\}) \neq \emptyset.$$

When this is true $h = 1_X \dagger [x \mapsto x']$, where 1_X denotes the identity function on the set X and $x' \in f^{-1}(\{y\})$. Thus, in general we can not use composition in these ways to construct the operation of override.

Override can be defined in terms of *removal* ($\triangleleft$) and *extension* ($\sqcup$). If $X \overset{\alpha}{\rightsquigarrow} Y$ and $X \overset{\beta}{\rightsquigarrow} Y$ are two partial functions between the sets X and Y then

$$\alpha \dagger \beta = \triangleleft_{\text{dom } \beta} \alpha \sqcup \beta.$$

Hence, override depends on the *shape* on X , the poset $\mathcal{P}X$.

Removal depends also on the *shape* on X .

We are thus led deeper into category theory to find a solution . . .

2 Categorical Preliminaries

Category theory (Lane 1971) was founded by S. Eilenberg and S. Mac Lane in 1945 and came from their work in algebraic topology. Today category theory is influencing many parts of theoretical computer science (Barr and Wells 1995; Walters 1991). This section is inspired by Robert Goldblatt's (Goldblatt 1984) exceptional book.

In set theory, the most primitive concept is *element*. Sets are built out of elements, also tuples, functions, and other mathematical structures are just different kinds of sets. In category theory the most primitive concept is *arrow*. Each arrow has a source and target *object*. The internal structure of these objects is ignored. The properties of an arrow are described in terms of composition with other arrows.

A *category* $\mathcal{C}$ comprises

1. a collection of things called $\mathcal{C}$ -objects;
2. a collection of things called $\mathcal{C}$ -arrows;
3. operations assigning to each $\mathcal{C}$ -arrow f a $\mathcal{C}$ -object $\text{dom } f$ the domain of f and a $\mathcal{C}$ -object $\text{cod } f$ the codomain of f . If $a = \text{dom } f$ and $b = \text{cod } f$ then we display this as

$$f: a \rightarrow b \text{ or } a \xrightarrow{f} b;$$

4. an operation assigning to each pair $\langle f, g \rangle$ of $\mathcal{C}$ -arrows with $\text{dom } g = \text{cod } f$, a $\mathcal{C}$ -arrow $g \circ f$, the *composite of f and g* , having $\text{dom}(g \circ f) = \text{dom } f$ and $\text{cod}(f \circ g) = \text{cod } g$, and such that the following condition obtains:

Associative Law: Given the configuration

$$a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} d$$

of $\mathcal{C}$ -objects and $\mathcal{C}$ -arrows then $h \circ (g \circ f) = (h \circ g) \circ f$.

5. an assignment to each $\mathcal{C}$ -object b of a $\mathcal{C}$ -arrow $1_b: b \rightarrow b$, called the *identity arrow on b* , such that

Identity Law: For any $\mathcal{C}$ -arrows $f: a \rightarrow b$ and $g: b \rightarrow c$

$$1_b \circ f = f, \text{ and } g \circ 1_b = g.$$

The category **Set** has sets as objects and total functions between sets as arrows. Composition of arrows is set-theoretic function composition. Identity arrows are identity functions.

The category $\text{Set}^{\downarrow}$ of functions has as objects the set functions $\begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array}$. An arrow from the $\text{Set}^{\downarrow}$ -object $\begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array}$ to the $\text{Set}^{\downarrow}$ -object $\begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array}$, denoted $\begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array} \xrightarrow{\langle h, k \rangle} \begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array}$, is a pair of functions $\langle h, k \rangle$ such that $\begin{array}{ccc} \begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array} & \xrightarrow{h} & \begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array} \\ & \xrightarrow{k} & \\ & \begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array} & \end{array}$ commutes, that is, $g \circ h = k \circ f$. For composition we put $\langle j, l \rangle \circ \langle h, k \rangle = \langle j \circ h, l \circ k \rangle$. The identity arrow for the $\text{Set}^{\downarrow}$ -object $\begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array}$ is the function pair $\langle 1_{X_0}, 1_{X_1} \rangle$.

2.1 Monic Arrows

An arrow $f: a \rightarrow b$ in a category $\mathcal{C}$ is *monic* if for any pair $g: c \rightarrow a$, $h: c \rightarrow a$ of $\mathcal{C}$ -arrows, the equality $f \circ g = f \circ h$ implies that $g = h$. The symbolism $f: a \rightarrowtail b$ is used to indicate that f is monic.

In the category **Set**, the monic arrows are just the *injective* functions.

A $\text{Set}^{\downarrow}$ -arrow $\begin{array}{ccc} \begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array} & \xrightarrow{\langle h, k \rangle} & \begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array} \end{array}$ between $\text{Set}^{\downarrow}$ -objects $\begin{array}{c} X_0 \\ f \downarrow \\ X_1 \end{array}$, $\begin{array}{c} Y_0 \\ g \downarrow \\ Y_1 \end{array}$ is monic if the functions $h: X_0 \rightarrow Y_0$, $k: X_1 \rightarrow Y_1$ are injective functions.

2.2 Iso Arrows

A $\mathcal{C}$ -arrow $f: a \rightarrow b$ is *iso*, or *invertible*, in $\mathcal{C}$ if there is a $\mathcal{C}$ -arrow $g: b \rightarrow a$, such that $g \circ f = 1_a$ and $f \circ g = 1_b$.

In the category **Set**, the iso arrows are just the *bijective* functions.

2.3 Isomorphic Objects

Object a and b are *isomorphic* in $\mathcal{C}$ denoted $a \cong b$, if there is a $\mathcal{C}$ -arrow $f: a \rightarrow b$ that is iso in $\mathcal{C}$.

2.4 Initial Object

An object 0 is *initial* in category $\mathcal{C}$ if for every $\mathcal{C}$ -object a there is one and only one arrow from 0 to a in $\mathcal{C}$.

In the category **Set** the empty set $\emptyset$, also denoted $\mathbf{0}$, is the unique initial object: for every set S , the empty function is the only function from $\mathbf{0}$ to S .

2.5 Terminal Object

An object 1 is *terminal* in a category $\mathcal{C}$ if for every $\mathcal{C}$ -object a there is one and only one arrow from a to 1 in $\mathcal{C}$.

Each one-element set is a terminal object in **Set**, since for every for every set S there is a function from S to a one-element set $\{0\}$, also denoted $\mathbf{1}$, mapping every element of S to 0 , and this is the only total function from S to $\mathbf{1}$.

2.6 Products

A *product* in a category $\mathcal{C}$ of two objects a and b is a $\mathcal{C}$ -object $a \times b$ together with a pair $pr_a: a \times b \rightarrow a$, $pr_b: a \times b \rightarrow b$ of $\mathcal{C}$ -arrows, called *projection* arrows, such that for *any* pair of $\mathcal{C}$ -arrows of the form $f: c \rightarrow a$, $g: c \rightarrow b$ there is exactly one arrow $\langle f, g \rangle: c \rightarrow a \times b$ making

commute.

Dashed arrows in commutative diagrams are used to represent arrows that are *asserted to exist uniquely* when the rest of the diagram is filled in appropriately.

In the category **Set**, the product of two sets X and Y is there *cartesian product* set

$$X \times Y = \{\langle x, y \rangle : x \in X \text{ and } y \in Y\}$$

with the pair of *projection functions*

$$pr_X: X \times Y \rightarrow X \text{ and } pr_Y: X \times Y \rightarrow Y$$

given by the rules

$$pr_X(\langle x, y \rangle) = x \text{ and } pr_Y(\langle x, y \rangle) = y.$$

2.7 Co-products

A *co-product* of $\mathcal{C}$ -objects a and b is a $\mathcal{C}$ -object $a + b$ together with a pair $i_a: a \rightarrow a + b, i_b: b \rightarrow a + b$ of $\mathcal{C}$ -arrows, called *injection arrows*, such that for any pair of $\mathcal{C}$ -arrows of the form $f: a \rightarrow c, g: b \rightarrow c$ there is exactly one arrow $[f, g]: a + b \rightarrow c$ making

$$\begin{array}{ccccc} a & \xrightarrow{i_a} & a + b & \xleftarrow{i_b} & b \\ & \searrow f & \downarrow [f, g] & \swarrow g & \\ & & c & & \end{array}$$

commute.

In $\mathbf{Set}$, the co-product of X and Y is their *disjoint union*

$$X \uplus Y = (X \times \{0\}) \cup (Y \times \{1\})$$

with *injection functions*

$$i_X: X \rightarrow X \uplus Y \text{ and } i_Y: Y \rightarrow X \uplus Y$$

given by the rules

$$i_X(x) = \langle x, 0 \rangle \text{ and } i_Y(y) = \langle y, 1 \rangle.$$

2.8 Pullback

A *pullback* of a pair $f: a \rightarrow c, g: b \rightarrow c$ of $\mathcal{C}$ -arrows with a common codomain is a pair of $\mathcal{C}$ -arrows $f': d \rightarrow a, g': d \rightarrow b$ such that

1. $f \circ g' = g \circ f'$, and
2. whenever $j: e \rightarrow b$, $h: e \rightarrow a$ are such that $f \circ h = g \circ j$, then

there is exactly one $\mathcal{C}$ -arrow $k: e \rightarrow d$ such that $h = g' \circ k$ and $j = f' \circ k$.

In **Set**, the pullback

of two set functions $f: X \rightarrow Z$ and $g: Y \rightarrow Z$ is defined by putting

$$X \times_Z Y = \{ \langle x, y \rangle : x \in X, y \in Y, \text{ and } f(x) = g(y) \}$$

with f' and g' as the projections

$$f'(\langle x, y \rangle) = x \text{ and } g'(\langle x, y \rangle) = y.$$

The pullback of $\mathbf{Set}^{\downarrow}$ -arrows can be found in (Goldblatt 1984, page 86).

2.9 Exponentiation

A category $\mathcal{C}$ has *exponentiation* if it has a product for any two $\mathcal{C}$ -objects, and if for any given $\mathcal{C}$ -objects a and b there is a $\mathcal{C}$ -object b^a and a $\mathcal{C}$ -arrow $ev: b^a \times a \rightarrow b$, called an *evaluation* arrow, such that for any $\mathcal{C}$ -object c and $\mathcal{C}$ -arrow $g: c \times a \rightarrow b$, there is a unique $\mathcal{C}$ -arrow $\hat{g}: c \rightarrow b^a$ making

commute.

In **Set**, the exponentiation of the sets X and Y is the set Y^X of all functions with domain X and codomain Y

$$Y^X = \{ f: X \rightarrow Y \}$$

with the evaluation function $ev: Y^X \times X \rightarrow Y$ given by the rule

$$ev(\langle f, x \rangle) = f(x).$$

The exponentiation of $\mathbf{Set}^{\downarrow}$ -objects can be found in (Goldblatt 1984, page 88).

2.10 Cartesian Closed Category

A *cartesian closed category* $\mathcal{C}$ is a category with

1. a terminal object 1 .
2. products, that is, a product object $a \times b$ for every pair of $\mathcal{C}$ -objects a and b .
3. exponentiation.

The categories **Set** and $\mathbf{Set}^{\downarrow}$ are cartesian closed.

2.11 Subobjects

The concept of *subobject* or *part* of a $\mathcal{C}$ -object d has two ingredients:

1. a part has a *shape*, a $\mathcal{C}$ -object a .
2. a monic arrow $f: a \rightarrow d$ from the shape of the part to the object d which determines how the shape a is *inserted* into the object d .

A subobject of a set X in the category **Set** with shape S is an injective function $f: S \rightarrow X$. This function determines a *subset* of the set X the *image* of f by $\text{Im } f = \{f(x) : x \in S\}$. The function f induces a bijection between S and $\text{Im } f$, so $S \cong \text{Im } f$. There is an *inclusion* function from the image of f into the set X , denoted $\text{Im } f \hookrightarrow X$. Thus in **Set** a subobject of a set X can be identified with a subset of X and an inclusion function from this subset into the set X .

injective functions we can identify them with inclusions, so that $S_0 \subset X_0$, $S_1 \subset X_1$ and f is the restriction of g , that is, $f(x) = g(x)$ for $x \in S_0$.

2.12 Subobject Classifier

If $\mathcal{C}$ is a category with terminal object 1 , then a *subobject classifier* or *truth-value object* for $\mathcal{C}$ is a $\mathcal{C}$ -object Ω together with a $\mathcal{C}$ -arrow $true: 1 \rightarrow \Omega$ that satisfies the following axiom.

Ω -axiom. For each monic $f: a \rightarrow d$ there is one and only one $\mathcal{C}$ -arrow $\chi_f: d \rightarrow \Omega$ such that

$$\begin{array}{ccc} a & \xrightarrow{f} & d \\ \downarrow & & \downarrow \chi_f \\ 1 & \xrightarrow{true} & \Omega \end{array}$$

is a pullback square.

If the set S is a subobject of the set X in **Set** then we have the inclusion $S \hookrightarrow X$. An element x of X can be classified now in two ways. Either

1. $x \in S$, or

2. $x \notin S$.

So we introduce the 2-element set $\{0, 1\}$, denoted **2**, and define $\chi_S: X \rightarrow \mathbf{2}$ as

$$\chi_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{if } x \notin S \end{cases}$$

We can now form the square

$$\begin{array}{ccc} S & \hookrightarrow & X \\ \downarrow & & \downarrow \chi_S \\ \mathbf{1} & \xrightarrow{\text{true}} & \mathbf{2} \end{array}$$

where $\text{true}(0) = 1$. The subobject classifier for **Set** is the function true from **1** to **2**. The diagram is a pullback square in **Set**. Thus, a part $S \hookrightarrow X$ of a set X has a characteristic function $\chi_S: X \rightarrow \mathbf{2}$.

Consider the category $\mathbf{Set}^{\downarrow}$. A part of an object $\begin{array}{c} X_0 \\ \downarrow f \\ X_1 \end{array}$ will be the inclusion $\begin{array}{ccc} S_0 & \hookrightarrow & X_0 \\ \downarrow h & & \downarrow f \\ S_1 & \hookrightarrow & X_1 \end{array}$ where

$S_0 \subseteq X_0$, $S_1 \subseteq X_1$, and $h(x) = f(x)$ for $x \in S_0$. An element x of X_0 can be classified now in three ways. Either

1. $x \in S_0$, or
2. $x \notin S_0$, but $f(x) \in S_1$, or
3. $x \notin S_0$, but $f(x) \notin S_1$.

So we introduce a 3-element set $\{0, 1, 2\}$, denoted **3**, and define $\Psi: X_0 \rightarrow \mathbf{3}$ by

$$\Psi(x) = \begin{cases} 2 & \text{if 1. holds} \\ 1 & \text{if 2. holds} \\ 0 & \text{if 3. holds} \end{cases}$$

We can now form the cube

$$\begin{array}{ccccc} S_0 & \xhookrightarrow{i_0} & X_0 & & \\ \downarrow h & \searrow & \downarrow f & & \\ & S_1 & \xhookrightarrow{i_1} & X_1 & \\ \downarrow & \downarrow \Psi & & \downarrow \chi_{S_1} & \\ \mathbf{1} & \xrightarrow{t'} & \mathbf{3} & \xrightarrow{t} & \mathbf{2} \\ \downarrow 1_1 & & & & \downarrow \text{true} \\ \mathbf{1} & \xrightarrow{\text{true}} & \mathbf{2} & & \end{array}$$

where $\text{true}(0) = 1$, $t'(0) = 2$ and where $t(2) = t(1) = 1$ and $t(0) = 0$. The base of the cube displays the

subobject classifier $\begin{array}{c} \mathbf{1} \\ \downarrow 1_1 \\ \mathbf{1} \end{array} \xrightarrow{\langle t', \text{true} \rangle} \begin{array}{c} \mathbf{3} \\ \downarrow t \\ \mathbf{2} \end{array}$ for $\mathbf{Set}^{\downarrow}$, which is the pair $\langle t', \text{true} \rangle$. The front and back faces of the cube

are each pullbacks in **Set**. The whole diagram exhibits $\begin{array}{c} X_0 \\ \downarrow f \\ X_1 \end{array} \xrightarrow{\langle \Psi, \chi_{S_1} \rangle} \begin{array}{c} 3 \\ \downarrow t \\ 2 \end{array}$ as the character in $\mathbf{Set}^{\downarrow}$ of the part

$$\begin{array}{c} S_0 \\ \downarrow h \\ S_1 \end{array} \xrightarrow{\langle i_0, i_1 \rangle} \begin{array}{c} X_0 \\ \downarrow f \\ X_1 \end{array}, \text{ see (Fig. 1).}$$

Figure 1: A character of a part.

2.13 Topoi

It is a category whose structure is sufficiently like **Set** that in it the interpretations of basic set-theoretical constructions behave much as they do in **Set** itself.

Goldblatt (Goldblatt 1984)

A category $\mathcal{C}$ is a *topos* if

1. $\mathcal{C}$ is cartesian closed.
2. $\mathcal{C}$ has a subobject classifier $1 \rightarrow \Omega$.

The categories **Set** and $\mathbf{Set}^{\downarrow}$ are topoi.
Why is topos theory useful to us?

- The ‘natural logic’ in a topos is intuitionistic logic.
- Intuitionistic logic fits naturally with the School of Constructive Mathematics.
- There is no conflict with ‘classical continuous’ Mathematics.

What more could we ask for in Computer Science?

3 Towards Override on Partial Arrows between Objects in a Topos

We wish to generalize the basic operations: extend, removal, restriction and override to topoi. To achieve this we must construct each of the operations in the topos **Set** and use these constructions as the definitions of the operations in a topos $\mathcal{E}$. In doing this we will have given a categorical semantics to the operations.

3.1 Partial Functions and Partial Arrows

In the topos **Set** a *partial function* f from a set A to a set B , written $f: A \rightsquigarrow B$, is a function from a subset of A to B i.e. $\text{dom } f \hookrightarrow A$ and $f: \text{dom } f \rightarrow B$. In a topos $\mathcal{E}$ a *partial arrow* f from an object a to an object b , written $f: a \rightsquigarrow b$, is a $\mathcal{E}$ -arrow $f: \text{dom } f \rightarrow b$, and there is a $\mathcal{E}$ -monic $\text{dom } f \hookrightarrow a$, see Goldblatt (Goldblatt 1984, page 267).

3.2 Extend in Set and $\mathcal{E}$

Given two partial functions $f: A \rightsquigarrow B$ and $g: A \rightsquigarrow B$ from a set A to a set B in the topos **Set**, where the two partial functions have disjoint domains, i.e. $\text{dom } f \cap \text{dom } g = \emptyset$, then we can *extend* the partial function f by the partial function g , written $f \sqcup g: A \rightsquigarrow B$. This extension is the pair of co-product functions $[f, g]: \text{dom } f \sqcup \text{dom } g \rightarrow B$ of functions f and g , and also $[i_1, i_2]: \text{dom } f \sqcup \text{dom } g \rightarrow A$ of the inclusion function $i_1: \text{dom } f \hookrightarrow A$ and $i_2: \text{dom } g \hookrightarrow A$. The co-product in **Set** is the *disjoint union* of two sets, but $\text{dom } f \sqcup \text{dom } g \cong \text{dom } f \cup \text{dom } g$ when $\text{dom } f \cap \text{dom } g = \emptyset$. Hence, when we have the disjointness condition the co-product diagram in **Set** becomes

We would like to define in a general topos $\mathcal{E}$ the extension of one partial arrow $f: a \rightsquigarrow b$ by another partial arrow $g: a \rightsquigarrow b$. To achieve this we must generalize the disjoint domains condition to a topos. In **Set** the diagram

is a pullback square precisely when $\text{dom } f \cap \text{dom } g = \emptyset$. Thus, in a general topos $\mathcal{E}$ we define the disjointness condition to be when the diagram

is a pullback square in $\mathcal{E}$, see Goldblatt (Goldblatt 1984, page 199). Given two partial arrows $f: a \rightsquigarrow b$ and $g: a \rightsquigarrow b$ with this condition we define their extension, written $f \sqcup g: a \rightsquigarrow b$, to be the pair of co-product arrows $[f, g]: \text{dom } f + \text{dom } g \rightarrow b$ of arrows f and g , and also $[m, n]: \text{dom } f + \text{dom } g \rightarrow a$ of subobjects $m: \text{dom } f \hookrightarrow a$ and $n: \text{dom } g \hookrightarrow a$, which are given by the partial arrows $f: a \rightsquigarrow b$ and $g: a \rightsquigarrow b$

3.3 Restrict in Set and $\mathcal{E}$

In **Set** if we have a partial function $f: A \rightsquigarrow B$ from a set A to a set B and we have a subset S of A then we can *restrict* the partial function f by the subset S of A , written $\triangleleft_S f: A \rightsquigarrow B$. This restriction is the pair of composite functions j, k in the diagram

If in a topos $\mathcal{E}$ we have a partial arrow $f: a \rightsquigarrow b$ from an object a to an object b and we have a subobject $n: s \rightarrow a$ of object a , then we can form the *intersection* of the subobjects $m: \text{dom } f \rightarrow a$ and $n: s \rightarrow a$. To do this we form the pullback of $m: \text{dom } f \rightarrow a$ and $n: s \rightarrow a$

The composite arrow $m \cap n = m \circ j = n \circ k$ is the intersection of the subobjects m and n . We can then define the restriction of the partial arrow $f: a \rightsquigarrow b$ by the subobject $n: s \rightarrow a$, written $\triangleleft_n f: a \rightsquigarrow b$, to be the pair of composite arrows in the diagram

3.4 Removal in Set and $\mathcal{E}$

In **Set** if we have a partial function $f: A \rightsquigarrow B$ from a set A to a set B and we have a subset S of A then we can *remove* the subset S of A from the partial function f , written $\triangleleft_S f: A \rightsquigarrow B$. This removal is the pair of composite functions j, k in the diagram

How can the set $\text{dom } f \setminus S$ be characterized? The set $\text{dom } f \setminus S$ is the only subset of the set $\text{dom } f$ which satisfies the equations

$$S \cap \text{dom } f \setminus S = \emptyset \tag{1}$$

$$(\text{dom } f \setminus S) \uplus (\text{dom } f \cap S) \cong \text{dom } f \tag{2}$$

These equations must be presented categorically if we wish to generalize the operation of removal to a topos.

If in a topos $\mathcal{E}$ we have a partial arrow $f: a \rightsquigarrow b$ from an object a to an object b and we have a subobject $n: s \rightarrowtail a$ of object a , then to form the removal of the subobject $n: s \rightarrowtail a$ from the partial arrow $f: a \rightsquigarrow b$, written $\triangleleft_n f: a \rightsquigarrow b$, we must find a subobject $l: \text{dom } f \setminus s \rightarrowtail \text{dom } f$ of the object $\text{dom } f$ which has two properties:

1. composing the subobject $m: \text{dom } f \rightarrowtail a$, which is given by the partial arrow $f: a \rightsquigarrow b$, after the required subobject $l: \text{dom } f \setminus s \rightarrowtail \text{dom } f$ to form a subobject $m \circ l: \text{dom } f \setminus s \rightarrowtail a$ of a , this subobject must be disjoint from subobject $n: s \rightarrowtail a$, i.e. the diagram

$$\begin{array}{ccc} 0 & \longrightarrow & \text{dom } f \setminus s \\ \downarrow & & \downarrow m \circ l \\ s & \xrightarrow{n} & a \end{array}$$

must be a pullback square in $\mathcal{E}$.

2. the co-product arrow $[l, j]: \text{dom } f \setminus s + \text{dom } f \cap s \rightarrowtail \text{dom } f$ of the required subobject $l: \text{dom } f \setminus s \rightarrowtail \text{dom } f$ of object $\text{dom } f$ and the subobject $j: \text{dom } f \cap s \rightarrowtail \text{dom } f$, formed from the intersection of the subobjects $m: \text{dom } f \rightarrowtail a$ and $n: s \rightarrowtail a$, see above, must be an $\mathcal{E}$ -iso, i.e. the arrow $[l, j]$ from the diagram

$$\begin{array}{ccccc} \text{dom } f \setminus s & \xrightarrow{\quad} & \text{dom } f \setminus s + \text{dom } f \cap s & \xleftarrow{\quad} & \text{dom } f \cap s \\ & \searrow l & \downarrow [l, j] & \swarrow j & \\ & & \text{dom } f & & \end{array}$$

must be an iso arrow in $\mathcal{E}$.

If such a subobject $l: \text{dom } f \setminus s \rightarrowtail \text{dom } f$ exists then we can define the removal of the subobject $n: s \rightarrowtail a$ from the partial arrow $f: a \rightsquigarrow b$, to be the pair of composite arrows in the diagram

$$\begin{array}{ccccc} & & f \circ l & & \\ & \text{dom } f \setminus s & \xrightarrow{l} & \text{dom } f & \xrightarrow{f} b \\ & \searrow m \circ l & & \downarrow m & \\ & & & a & \end{array}$$

3.5 Override in Set and $\mathcal{E}$

Given two partial functions $f: A \rightsquigarrow B$ and $g: A \rightsquigarrow B$ from a set A to a set B in the topos **Set**, then we can *override* the partial function f by the partial function g , written $f \dagger g: A \rightsquigarrow B$. This is the partial function formed by the expression $\triangleleft_{\text{dom } g} f \sqcup g$. This expression is always defined in the topos **Set** because we can always form the removal of the subset $\text{dom } g$ of A from the partial function f , i.e. the partial function $\triangleleft_{\text{dom } g} f: A \rightsquigarrow B$.

If in a topos $\mathcal{E}$ we have two partial arrows $f: a \rightsquigarrow b$ and $g: a \rightsquigarrow b$ from an object a to an object b and we can form the removal of the subobject $n: \text{dom } g \rightarrowtail a$, which is given by the partial arrow $g: a \rightsquigarrow b$, from the partial arrow $f: a \rightsquigarrow b$, i.e. the partial arrow $\triangleleft_n f: a \rightsquigarrow b$, then we can override the partial arrow f by the partial arrow g , written $f \dagger g: a \rightsquigarrow b$, this is the extension of the partial arrow $\triangleleft_n f$ by the partial arrow g . We may perform this extension because the partial arrow $\triangleleft_n f$ will be disjoint from the partial arrow g by definition.

4 An Example

pair $\langle \alpha_0 \uparrow \bar{\beta}_0, \alpha_1 \uparrow \bar{\beta}_1 \rangle$ will be a $\mathbf{Set}^{\downarrow}$ -arrow.

5 Epilogue

The basic operations: extend, restriction, removal and override have been introduced to topoi and hence given a categorical semantics. There are many questions yet to answer:

- can override be defined by universal properties?
- in which topoi can a subobject $n: s \rightarrowtail a$ of topos object a be removed from a partial arrow $f: a \rightsquigarrow b$ between objects a and b ?
- if $\text{PA}(a, b)$ denotes the collection of all partial arrows $f: a \rightsquigarrow b$ between objects a and b , and override is defined for each pair of partial arrows in $\text{PA}(a, b)$ then does collection $\text{PA}(a, b)$ with the operation override form an algebraic structure? For example, in the topos **Set** the collection of partial functions between two sets with the operation override forms a monoid (Mac an Airchinnigh 1990, page 99),
- in the topos **Set** the operations have many inter-relationships, e.g. if S is a subset of set A and $f: A \rightsquigarrow B$ is a partial function between sets A and B then

$$f = \triangleleft_S f \sqcup \triangleleft_S f,$$

In which topoi do these relationships hold?

There are many directions now opened up for future development:

- the use of topoi other than **Set** for modelling systems,
- the implication of this work for the general theory of topoi,
- the introduction of the indexed operations (Mac an Airchinnigh 1993; Hughes and Donnelly 1995; Gallagher and Hughes 1996) to topoi.

5.1 Acknowledgments

Thanks goes to all the members of the *Formal Methods Working Group*, at the University of Dublin, Trinity College, who have been patient and tolerant enough to have endured much of the development described in the paper on many occasions. I would particularly like to thank Dr. Mícheál Mac an Airchinnigh for his constant encouragement, feedback and his **Set**[↓] macros.

I would also like to express my thanks to a diligent, but anonymous, reviewer who pointed out numerous errors which had crept into an earlier version. As is customary, I acknowledge that any errors of whatever form that remain in the paper are my sole responsibility.

The commuting diagrams in this paper were produced by the $\text{\TeX}$ package $\text{\Xy-pic}$ ¹ of Kristoffer H. Rose BRICS, Computer Science, Aarhus University (bld. 540), Ny Munkegade, DK-8000 Aarhus C, Denmark (e-mail: kris-rose@brics.dk).

The bibliography is produced with the aid of $\text{\MacBibTeX}$ ² of Vince Darley, Division of Applied Sciences, Harvard University, Oxford Street, Cambridge MA 02138, USA (e-mail: vince@das.harvard.edu).

References

- Barr, M. and C. F. Wells (1995). *Category Theory for Computing Science* (Second ed.). International series in computer science. Prentice Hall.
- Gallagher, A. D. D. and A. Hughes (1996, March). On the Inheritance of Monoid Properties in Indexed Structures, A Tale of Three Proofs. Technical report, Department of Computer Science, University of Dublin, Trinity College, Dublin, Ireland.
- Goldblatt, R. (1984). *Topoi: The Categorical Analysis of Logic*, Volume 98 of *Studies in Logic and the Foundations of Mathematics*. North-Holland.
- Hughes, A. and A. Donnelly (1995, September). An Algebraic Proof in $\text{VDM}^\clubsuit$. In J. Bowen and M. Hinchey (Eds.), *ZUM'95: The Z Formal Specification Notation*, Volume 967 of *Lecture Notes in Computer Science*, pp. 114 – 133. Springer-Verlag.
- Hughes, A. and A. Donnelly (1998). Algebraic Advances for Aliasing. In D. Duke and A. Evans (Eds.), *Northern Formal Methods Workshop, Ilkley 1997*, Electronic Workshops in Computing. Springer-Verlag. <http://www.springer.co.uk/ewic/workshops/>.
- Hughes, A. and C. Pahl (1997). A Generic Model for State-based Agent Systems. In G. O'Regan and S. Flynn (Eds.), *1st Irish Workshop on Formal Methods, Dublin 1997*, Electronic Workshops in Computing. Springer-Verlag. <http://www.springer.co.uk/ewic/workshops/>.
- Jones, C. (1990). *Systematic Software Development using VDM* (Second ed.). International Series in Computer Science. Prentice Hall.
- Lane, S. M. (1971). *Categories for the Working Mathematician*. Springer-Verlag.
- Lawvere, F. W. (1975). Variable sets etendu and variable structures in topoi. Technical report, Department of Mathematics, The University of Chicago.

¹<ftp://ftp.mpce.mq.edu.au/pub/mathsf/TeX/>

²<http://www.fas.harvard.edu/~darley/>

- Lawvere, F. W. and S. H. Schanuel (1997). *Conceptual Mathematics: A first introduction to categories*. Cambridge University Press.
- Mac an Airchinnigh, M. (1990). *Conceptual Models and Computing*. Ph. D. thesis, Department of Computer Science, University of Dublin, Trinity College, Dublin, Ireland.
- Mac an Airchinnigh, M. (1991). Tutorial Lecture Notes on the Irish School of the VDM. In S. Prehn and W. J. Toetenel (Eds.), *VDM'91: Formal Software Development Methods*, Volume 2 of *Lecture Notes in Computer Science*, pp. 141 – 237. Springer-Verlag.
- Mac an Airchinnigh, M. (1993). Formal Methods and Testing. In *Tutorial Notes: 6th International Software Quality Week*, Software Research Institute, 625 Third Street, San Francisco, CA 94107-1997.
- Poppleton, M. R. (1997). The single transferable voting system: Functional decomposition in formal specification. In G. O'Regan and S. Flynn (Eds.), *1st Irish Workshop on Formal Methods, Dublin 1997*, Electronic Workshops in Computing. Springer-Verlag. <http://www.springer.co.uk/ewic/workshops/>.
- Spivey, M. (1992). *The Z Notation* (Second ed.). International Series in Computer Science. Prentice Hall.
- Walters, R. (1991). *Categories and Computer Science*, Volume 28 of *Cambridge Computer Science Tests*. Cambridge University Press.