INFORMATIQUE THÉORIQUE ET APPLICATIONS

WAFAA KHALIL
R. F. C. WALTERS

An imperative language based on distributive categories II

Informatique théorique et applications, tome 27, nº 6 (1993), p. 503-522.

<http://www.numdam.org/item?id=ITA_1993__27_6_503_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 27, n° 6, 1993, p. 503 à 522)

AN IMPERATIVE LANGUAGE BASED ON DISTRIBUTIVE CATEGORIES II (*)

by Wafaa KHALIL (1) and R. F. C. WALTERS (1)

Communicated by G. LONGO

Abstract. – *This paper continues the analysis of the imperative languages, $IMP(\mathbf{G})$, begun in Walters* [1, 2, 3]. *We describe a precise syntax and some programming techniques. The programming techniques are based on the simple and important notion of a* functional processor.

As an illustration of programming in these languages we give a universal $IMP(\mathbf{G})$ program written in $IMP(\hat{\mathbf{G}})$, where $\hat{\mathbf{G}}$ is an extension of $\mathbf{G}$ *by certain stack types.*

Résumé. – *Cet article poursuit l'analyse des langages impératifs, $IMP(\mathbf{G})$, entreprise dans Walters* [1, 2, 3]. *Nous décrivons une syntaxe précise et quelques techniques de programmation. Ces techniques reposent sur une notion simple et importante de* processeur fonctionnel.

Comme exemple de programmation dans ces langages, nous donnons un programme $IMP(\mathbf{G})$ universel écrit en $IMP(\hat{\mathbf{G}})$, où $\hat{\mathbf{G}}$ est une extension de $\mathbf{G}$ à certains types de piles.

1. INTRODUCTION

In Walters [1, 2, 3], the second author described a family of imperative languages based on iteration and the operations of a distributive category. These will be revised in this paper. Each language in the family depends on a suitable graph $\mathbf{G}$ of given functions; we will denote the language corresponding to the graph $\mathbf{G}$ by IMP($\mathbf{G}$). The language IMP($\mathbf{G}$) is abstract and mathematically based with no prescribed control strategy. An *isolated* program, P, is just a function, $\mathrm{act}_P : X_P \to X_P$, built out of the given functions using the operations of a distributive category. The function act_P *is called the action* of P, and X_P the *state space* of P.

In paragraph 2 we begin by describing some programming methods, the main tool is that of *functional processor* or *pseudofunction*.

(*) Received November 1991, accepted December 1991.

(1) School of Mathematics and Statistics, University of Sydney, N.S.W. 2006, Australia.

There is no precise syntax for IMP (G) given in [1, 2, 3]. In order to avoid an *ad hoc* choice of syntax, (the text of) a program was taken there to be a loop in a free distributive category – that is, it can be taken to be a certain equivalence class of strings. In this paper we must take the plunge and decide on a specific syntax. This is done in paragraph 3, where we also define the operation of a program. We show there that any program has the same behaviour as one which is a composite of certain *elementary* arrows. We associate a *token* with each elementary arrow such that the effect of each elementary arrow is a simple string manipulation determined by the token of the arrow.

The remainder of the paper is concerned with constructing a universal IMP (G) program, $\mathscr{U}$, written in IMP ($\hat{\mathbf{G}}$) where $\hat{\mathbf{G}}$ is an extension of **G** by certain stack types.

The state space of the universal program is of the form

$$X_{\mathscr{U}} = S_{\text{char}} \times S_{\text{data}} + Z$$

where S_{char} is a type stack of characters, S_{data} is a type stack of data elements, and Z is an unspecified set – of *local states* of the program. We call the states of $X_{\mathscr{U}}$, that are in the component $S_{\text{char}} \times \text{S}_{\text{data}}$ the *global states* of $\mathscr{U}$.

The universal program *implements* each isolated IMP (G) program, P, in the following precise sense. Suppose the initial state of $\mathscr{U}$ is a global state (t, x_0), where t is the program text of P and x_0 is a suitable initial state of P. The sequence of global states of $\mathscr{U}$ under the iteration of $\text{act}_{\mathscr{U}}$ is then

$$(t, x_0), \quad (t, x_1), \quad (t, x_2), \ldots$$

where $x_0, x_1, x_2, \ldots$ is the behaviour of P with initial state x_0.

Since the language IMP (G) is mathematically based, it is straight forward to prove the behaviour of the universal program. Another consequence of the mathematical basis of the language IMP (G) is that there are various aspects of this paper that are of mathematical as well as computational interest. For example, we use the fact that any set built up out of the sets $A, B, \ldots$ using product and sums maybe represented as a *subset* of $(A+B+\ldots+I)^*$, then the associativity isomorphisms for sums and products (but not the distributivity isomorphisms) are identities. This allows $\hat{\mathbf{G}}$ to be an extension of **G** by stack types.

The authors are grateful for helpful suggestions given by Michael Johnson and Eric Wagner. We also wish to thank Paul Taylor for the use of "Paul

Taylor's Commutative diagrams". The second author's research was supported by an Australian Research Council Program Grant, and an Australian Research Council Small Grant.

2. PROGRAMMING METHODS

In this section we develop some general methods for constructing programs in the IMP family of languages. They are precisely the tools we need in the construction of a universal IMP (G) program. First we introduce the concept of a *pseudofunction* from X to Y. For the notation used in this section *see* Walters ([1, 2]).

2.1. Pseudofunctions

DEFINITION 2.1: A program $\varphi : X+U+Y \to X+U+Y$ is said to idle in Y if $\varphi \circ j = j$, where j is the injection $j: Y \to X+U+Y$; that is $\varphi(y)=y$ if $y \in Y$.

A *pseudofunction* or *functional processor*, φ, from a set X to a set Y, denoted $\varphi : X \nrightarrow Y$, is a program $\varphi : X+U+Y \to X+U+Y$ which idles in Y and with the property that for each $x \in X$ there exists a natural number n_x such that $\varphi^{n_x}(x) \in Y$.

The set U will be referred to as the set of *local states* of the pseudofunction φ.

PROPOSITION 2.1: *Let $\varphi : X \nrightarrow Y$ be a pseudofunction, then there is a function $\bar{\varphi} : X \to Y$ defined by $\bar{\varphi}(x) = \varphi^{n_x}(x)$, the function obtained by iterating φ.*

Proof: $\bar{\varphi}$ is fully defined because for each $x \in X$ there exists a natural number n_x such that $\varphi^{n_x}(x) \in Y$, and is single valued because φ idles in Y. □

PROPOSITION 2.2: *For any function $f : X \to Y$, let j be the injection $j: Y \to X+Y$, then the program $f' = \nabla_{X+Y} \circ (j \circ f + j) : X+Y \to X+Y$ is a pseudofunction such that $\bar{f}' = f$.*

Proof: f' is fully defined because $f'(x) = f(x) \in Y$, and it is easily checked that

$$\begin{aligned} f' :\ & X+Y \to X+Y \\ & (x, 0) \mapsto (f(x), 1) \\ & (y, 1) \mapsto (y, 1) \end{aligned}$$

therefore, f' idles on Y. □

In the following two examples, let *predecessor*: $\mathbb{N} \to I + \mathbb{N}$, *successor*: $I + \mathbb{N} \to \mathbb{N}$, *multiply*: $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and *difference*: $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be given functions, (where *difference* is the function which maps (m, n) to $|m-n|$).

Example 2.1: A pseudofunction, *factorial*: $\mathbb{N} \nrightarrow \mathbb{N}$, which calculate $n!$ for each $n \in \mathbb{N}$ is the program:

$$
\begin{aligned}
\textit{factorial:}\quad & \mathbb{N} + \mathbb{N}^2 + \mathbb{N} \to \mathbb{N} + \mathbb{N}^2 + \mathbb{N} \\
& (x, 0) \mapsto ((1, x), 1) \\
& ((p, m), 1) \mapsto \begin{cases} ((pm, m-1), 1), & \text{if } m \geqq 1 \\ (p, 2). & \text{if } m = 0 \end{cases} \\
& (n, 2) \mapsto (n, 2).
\end{aligned}
$$

For an indication of the way this program and the next are constructed, using the operations of a distributive category from given functions *see* Walters [1] and [2]. Note that to indicate when $x \in X$ belongs to the ith component of a sum, we write (x, i).

It can be checked that $\overline{factorial}(n) = n!$.

Example 2.2: A pseudofunction, $gcd: \mathbb{N} \times \mathbb{N} \nrightarrow \mathbb{N}$, which calculates the greatest common divisor for each pair $(m, n) \in \mathbb{N} \times \mathbb{N}$ is the program:

$$
\begin{aligned}
gcd:\quad & \mathbb{N}^2 + \mathbb{N} \to \mathbb{N}^2 + \mathbb{N} \\
& ((m, n), 0) \mapsto \begin{cases} ((n, |m-n|), 0), & \text{if } m > 0 \text{ and } n > 0 \\ (m+n, 1), & \text{if } m = 0 \text{ or } n = 0 \end{cases} \\
& (x, 1) \mapsto (x, 1).
\end{aligned}
$$

It can be checked that $\overline{gcd}(m, n) = gcd(m, n)$.

Note 2.1: The *factorial* program has local states in $\mathbb{N}^2$, while the *gcd* program has no local states.

2.2. Composition of pseudofunctions

PROPOSITION 2.3: *If* $\alpha: X \nrightarrow Y$ *and* $\beta: Y \nrightarrow Z$ *are pseudofunctions, with local states in* U *and* V *respectively, then*

$$\alpha; \beta = (1_{X+U} + \beta) \circ (\alpha + 1_{V+Z}) : X + W + Z \to X + W + Z$$

is a pseudofunction from X to Z, with local states in $W = U + Y + V$, and with the property that:

$$\overline{\alpha; \beta} = \bar{\beta} \circ \bar{\alpha}.$$

Proof: If $z \in Z$, then $(\alpha; \beta)(z) = \beta(z) = z$. Hence, $\alpha; \beta$ idles on Z. To find n_x for each $x \in X$ such that $(\alpha; \beta)^{n_x}(x) = (\bar{\beta} \circ \bar{\alpha})(x)$, notice that there is a least n_1 such that $\alpha^{n_1}(x) = \bar{\alpha}(x)$. Similarly, there is a least n_2 such that $\beta^{n_2}(\bar{\alpha}(x)) = \bar{\beta}(\bar{\alpha}(x)) \in Z$, since $\bar{\alpha}(x) \in Y$. Taking $n_x = n_1 + n_2 - 1$, then

$$\begin{aligned}(\alpha; \beta)^{n_x}(x) &= (\alpha; \beta)^{n_2 - 1}((\alpha; \beta)^{n_1}(x)) \\ &= (\alpha; \beta)^{n_2 - 1}(\beta(\bar{\alpha}(x)) \\ &= \bar{\beta}(\bar{\alpha}(x)).\end{aligned}$$

Therefore, $\overline{\alpha; \beta} = \bar{\beta} \circ \bar{\alpha}$. □

Example 2.3: $\overline{gcd; factorial}(m, n) = (gcd(m, n))!$

2.3. Cases and parallel processes

PROPOSITION 2.4: *If $\varphi : X \nrightarrow Y$ and $\psi : X' \nrightarrow Y'$ are pseudofunctions with local states U and U' respectively, then*

(i) $\varphi \vee \psi = a^{-1} \circ (\varphi + \psi) \circ a$ *where*

$$a = (1_{X+U} + \text{twist}_{(X'+U'), Y} + 1_{Y'}) \circ (1_X + \text{twist}_{X', U} + 1_{U'+Y+Y'})$$

is a pseudofunction from $X + X'$ to $Y + Y'$ with local states in $W = U + U'$, and with the property that:

$$\overline{\varphi \vee \psi} = \bar{\varphi} + \bar{\psi}.$$

(ii) $\varphi \wedge \psi = b^{-1} \circ (\varphi \times \psi) \circ b$ *where*

$$b = \delta_2 \circ (\delta_0 + 1_{(U \times (X' + U' + Y')} + \delta_1)$$

where δ_0, δ_1, δ_2 are distributive law arrows, is a pseudofunction from $X \times X'$ to $Y \times Y'$ with local states in

$$W = (X \times (U' + Y')) + (U \times (X' + U' + Y')) + (Y \times (X' + U')),$$

and with the property that:

$$\overline{\varphi \wedge \psi} = \bar{\varphi} \times \bar{\psi}.$$

Proof: (i). — Note that the effect of a is given by:

$$\begin{array}{lll}(x, 0) \mapsto (x, 0), & (x', 1) \mapsto (x', 3), & (u, 2) \mapsto (u, 1), \\ (u', 3) \mapsto (u', 4), & (y, 4) \mapsto (y, 2), & (y', 5) \mapsto (y', 5),\end{array}$$

where $u \in U$, $u' \in U'$, $y \in Y$, $y' \in Y'$. Therefore, the functions a and a^{-1} amount to rearranging the elements in the sum in order to do $(\varphi + \psi)$. If $y \in Y$, then $(\varphi \vee \psi)(y) = a^{-1} \circ (\varphi + \psi) \circ a(y) = a^{-1} \circ \varphi(y) = a^{-1}(y) = y$, since φ idles on Y. If $y' \in Y'$, then $(\varphi \vee \psi)(y') = a^{-1} \circ (\varphi + \psi) \circ a(y') = a^{-1} \circ \psi(y') = a^{-1}(y') = y'$, since ψ idles on Y'. Hence, $\varphi \vee \psi$ idles on $Y + Y'$. Now let n_1 and n_2 be the least natural numbers such that $\varphi^{n_1}(x) = \bar{\varphi}(x)$ and $\psi^{n_2}(x') = \bar{\psi}(x')$, where $x \in X$, $x' \in X'$. Take $n_x = n$ where n is the larger of n_1 and n_2, then

$$(\varphi \vee \psi)^n(x) = (a^{-1} \circ (\varphi^n + \psi^n) \circ a)(x) = \begin{cases} \bar{\varphi}(x), & \text{if } x \in X \\ \bar{\psi}(x), & \text{if } x \in X' \end{cases}$$

therefore, $\overline{\varphi \vee \psi} = \bar{\varphi} + \bar{\psi}$.

(ii) If $(y, y') \in Y \times Y'$ then

$$(\varphi \wedge \psi)(y, y') = b^{-1} \circ (\varphi \times \psi)((y, 2), (y', 2)) = b^{-1}((y, 2), (y', 2)) = (y, y').$$

Therefore $(\varphi \wedge \psi)$ idles on $Y \times Y'$. Now let n be defined as in the case for sums, then:

$$(\varphi \wedge \psi)^n(x, x') = (b^{-1} \circ (\varphi^n \times \psi^n) \circ b)(x, x') = (\bar{\varphi}(x), \bar{\psi}(x'))$$

therefore, $\overline{\varphi \wedge \psi} = \bar{\varphi} \times \bar{\psi}$. $\square$

Example 2.4.

$$\begin{aligned} \overline{gcd \vee factorial}: \quad & \mathbb{N}^2 + \mathbb{N} \to \mathbb{N} + \mathbb{N} \\ & ((l, m), 0) \mapsto (gcd(l, m), 0) \\ & (n, 1) \mapsto (n!, 1), \\ \overline{gcd \wedge factorial}: \quad & \mathbb{N}^2 \times \mathbb{N} \to \mathbb{N} \times \mathbb{N} \\ & ((l, m), n) \mapsto (gcd(l, m), n!). \end{aligned}$$

COROLLARY 2.1: *For $i = 1, 2, \ldots, n$ if $\varphi_i : X_i \nrightarrow X$ and $\psi_i : X \nrightarrow X_i$ are pseudofunctions; then there exists pseudofunctions $\varphi : X_1 + X_2 + \ldots + X_n \nrightarrow X$ and $\psi : X \nrightarrow X_1 \times X_2 \times \ldots \times X_n$ such that*

$$\begin{aligned} &\bar{\varphi}(x) = \bar{\varphi}_i(x), \qquad \textit{if} \quad x \in X_i, \\ &\bar{\psi}(x) = (\bar{\psi}_1(x), \bar{\psi}_2(x), \ldots, \bar{\psi}_n(x)). \end{aligned}$$

Proof: Take

$$\begin{aligned} &\varphi = (\varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n); \nabla \\ &\psi = \Delta; (\psi_1 \wedge \psi_2 \wedge \ldots \wedge \psi_n). \end{aligned}$$

Then the result follows from the construction of $\bar{\varphi}$ and $\bar{\psi}$. $\square$

2.4. Iteration of a pseudofunction

PROPOSITION 2.5: *Given a pseudofunction* $\mu : X \nrightarrow X + Y$ *and a function* $\operatorname{ord}: X \to \mathbb{N}$, *such that if* $\mu(x) \in X$, *then* $\operatorname{ord}(\mu(x)) < \operatorname{ord}(x)$; *then there exists a pseudofunction* $\nu : X \nrightarrow Y$ *such that*

$$\bar{\nu}(x) = \bar{\mu}^{m_x}(x), \quad \textit{for some } m_x \leq \operatorname{ord}(x) + 1$$

Proof: Let U be the set of local states of the program μ, and take

$$\nu = (\nabla_X + i_{U,X} + 1_Y) \circ (1_X + \operatorname{twist}_{U,X} + 1_Y) \circ \mu$$

where the local states of ν are in $W = U + X$. It is clear that $\nu \circ j = j$. To show that there exists a natural number n_x such that $\nu^{n_x}(x) \in Y$, suppose $\operatorname{ord}(x) = k$, and that x is in the first component of the sum. There is a number $n_1 \in \mathbb{N}$, such that $n_1 > 0$ and $\nu^{n_1}(x) = \bar{\mu}(x) = x_1$ is in the first X or in Y because μ is a pseudofunction. If it is in X then $\operatorname{ord}(x_1) \leq k - 1$. Iterating ν, there is a number $n_2 \in \mathbb{N}$ such that $\nu^{n_2}(x_1) = \bar{\mu}(x_1) = \bar{\mu}^2(x) = x_2$ is in the first X or is in Y. If it is in X, then $\operatorname{ord}(x_2) \leq k - 2$. Continuing the iteration, we get a sequence $x_1, x_2, x_3, \ldots$ of elements of X of strictly decreasing order. Clearly, there exists an $i \leq \operatorname{ord}(x)$ such that

$$\nu^{n_1 + n_2 + \ldots + n_i}(x) = \bar{\mu}^i(x) = x_i \in X$$

but

$$\nu^{n_1 + n_2 + \ldots + n_i + n_{i+1}}(x) = \bar{\mu}^{i+1}(x) \in Y.$$

Take $m_x = i + 1$, hence the result. □

Example 2.5: A pseudofunction $gcd : \mathbb{N}^2 \nrightarrow \mathbb{N}$, may be constructed from the following pseudofunction (arising from a function), using proposition 2.5,

$$\begin{aligned} g:\quad & \mathbb{N}^2 \nrightarrow \mathbb{N}^2 + \mathbb{N} \\ (m, n) \mapsto & \begin{cases} ((n, |m-n|), 0), & \text{if } m > 0 \text{ and } n > 0 \\ (m+n, 1), & \text{if } m = 0 \text{ or } n = 0 \end{cases} \end{aligned}$$

if the function ord is taken to be:

$$\begin{aligned} \operatorname{ord}:\quad & \mathbb{N}^2 \to \mathbb{N} \\ (m, n) \mapsto & \begin{cases} 2m, & \text{if } m > n \\ 2n+1, & \text{if } m \leq n. \end{cases} \end{aligned}$$

Note 2.2: (i) The state space for the above *gcd* program is not the same as that given in example 2.2.

(ii) The given functions *multiply* and *difference*, used in the construction of the programs *factorial* and *gcd*, could have been constructed as pseudofunctions in terms of the given functions *predecessor* and *successor*, again using the techniques we have developed in this section.

PROPOSITION 2.6: *Given a pseudofunction* $\alpha : X \leftrightarrow X$, *the program*

$$\mathrm{iter}(\alpha): \quad X+V \to X+V,$$

constructed below, has the property that if $n_0, n_1, n_2, \ldots$ *is the sequence of natural numbers for which* $(\mathrm{iter}(\alpha))^{n_i}(x) \in X$, *then*

$$(\mathrm{iter}(\alpha))^{n_i}(x) = \bar{\alpha}^i(x).$$

Proof: Let U be the set of local states of the program α, and let $V = X+U+X$, the state space of α; then

$$\mathrm{iter}(\alpha) = b \circ (1_X + \alpha) \circ a: \quad X+V \to X+V$$

where

$$a = (1_X + \nabla_{X+U+X}) \circ (i_1 + 1_{X+U+X}),$$

$$b = (\nabla_X + i_0) \circ (1_X + \mathrm{twist}_{(X+U),\, X}),$$

where $i_0 : X+U \to X+U+X$, and $i_1 : X \to X+X+U+X$, $i_1(x) = (x, 1)$, are injections.

Notice that a takes $(x, 0)$ to $(x, 1)$ and leaves everything else fixed. While b moves $(x, 3)$ to $(x, 0)$ and leaves everything else fixed. Iterating the program $\mathrm{iter}(\alpha)$, control will reach the last component because of a and the pseudofunction α; once it is there it is passed back to the first component in $X+V$, via b. It is then clear that $(\mathrm{iter}(\alpha))^{n_i}(x) = \bar{\alpha}^i(x)$. □

3. THE SYNTAX AND OPERATION OF THE LANGUAGE

3.1. Distributive graphs and expressions

DEFINITION 3.1: Let $\mathcal{O}$ be a set of objects, usually denoted $A, B, C \ldots$ Then (*distributive*) *expressions of objects* are strings, or words, formed from the objects in $\mathcal{O}$ together with the symbols O $I+$ $\times$ ( ) by the following rules:

(i) The symbols O and I are expressions of objects.

(ii) The objects in $\mathcal{O}$ are expressions of objects.

(iii) If U and V are expressions of objects, then the strings $(U\times V)$ and $(U+V)$ are expressions of objects.

Objects of **G** will usually be denoted by the letters A, B, $C\ldots$, while expressions of objects will be denoted by the letters X, Y, Z, U, V, $W, \ldots$

DEFINITION 3.2: A *distributive graph* is a set $\mathcal{O}$ of objects and a set $\mathcal{A}$ of arrows, with the arrows having assigned domains and codomains which are expressions of objects.

Remark 3.1: The objects of a distributive graph **G** can be thought of as the given data types, while the arrows can be thought of as the given functions. For example $\mathbb{N}$ is the natural numbers while *predecessor*: $\mathbb{N}\to\mathbb{N}+I$ and *successor*: $\mathbb{N}+I\to\mathbb{N}$, are given functions.

DEFINITION 3.3: Let **G** be a distributive graph. (*Distributive*) *expressions* of arrows are strings of the form $\alpha : U\to V$, where U and V are expressions of objects and α is a string formed from expressions of objects, the arrows of **G**, and the symbols

$$1 \ p \ q \ i \ j \ \delta^{-1} \ ! \ ¡ \ \Delta \ \nabla \ + \ \times \ \circ \ , \quad (\ )$$

by the following rules:

(i) The arrows in **G** are expressions of arrows.

(ii) Let X, Y, Z be expressions of objects, then the following expressions:

$$\begin{aligned}
1_X&: \ X\to X\\
p_{X,Y}&: \ (X\times Y)\to X\\
q_{X,Y}&: \ (X\times Y)\to Y\\
i_{X,Y}&: \ X\to(X+Y)\\
j_{X,Y}&: \ Y\to(X+Y)\\
\delta^{-1}_{X,Y,Z}&: \ (X\times(Y+Z))\to((X\times Y)+(X\times Z))\\
!_X&: \ X\to I\\
¡_X&: \ O\to X\\
\Delta_X&: \ X\to(X\times X)\\
\nabla_X&: \ (X+X)\to X
\end{aligned}$$

are expressions of arrows.

(iii) If $\alpha : U\to V$ and $\beta : X\to Y$ are expressions of arrows, then the strings

$$(\alpha\times\beta): \quad (U\times X)\to(V\times Y)$$

and

$$(\alpha+\beta): \quad (U+X) \to (V+Y)$$

are expressions of arrows.

(iv) If $\alpha_i : X_i \to X_{i+1}$ for $i=1,2,\ldots,n$ are expressions of arrows with $\alpha_i \neq 1_{X_i}$, $(i=1,2,\ldots n)$, then the string

$$\alpha_n \circ \alpha_{n-1} \circ \ldots \circ \alpha_1 : \quad X_1 \to X_{n+1}$$

is an expression of arrows.

Arrows of **G** will usually be denoted by the letters f, g, $h \ldots$, while expressions of arrows will be denoted by the letters α, β, $\gamma \ldots$

3.2. The text of a program

The sets of expressions of objects and expressions of arrows form the objects and arrows of a category, denoted Expr(**G**), where composition of expressions of arrows $\alpha : X \to Y$ and $\beta : Y \to Z$ is defined as follows:

(i) If α and β are *not* identities, then their composition is $\beta \circ \alpha : X \to Z$.

(ii) If α is an identity, then the composition of α and β is $\beta : Y \to Z$. Similarly, if β is an identity, then the composition is $\alpha : X \to Y$.

Example 3.1: Here are some arrows in Expr(**G**):

(i) $\mathrm{twist}_{(X,Y)} = (q_{X,Y} \times p_{X,Y}) \circ \Delta_{(X \times Y)} : (X \times Y) \to (Y \times X)$.

(ii) $\mathrm{twist}_{(X,Y)} = \nabla_{(Y+X)} \circ (j_{Y,X} + i_{Y,X}) : (X+Y) \to (Y+X)$.

(iii) Let $\gamma^{-1}_{X,Y,Z} = (\mathrm{twist}_{(Z,X)} + \mathrm{twist}_{(Z,Y)}) \circ \delta^{-1}_{Z,X,Y} \circ \mathrm{twist}_{((X+Y),Z)}$, then:

$$\gamma^{-1}_{X,Y,Z} : \quad ((X+Y) \times Z) \to ((X \times Z) + (Y \times Z)).$$

(iv) $(1_X \times !_X) \circ \Delta_X : X \to (X \times I)$.

(v) Associativity arrow for sums is demonstrated by the following example. Let

$$\mathrm{assoc} = \nabla_{(X+(Y+Z))} \circ (1_{(X+(Y+Z))} + (1_X + j_{Y,Z})) \circ ((1_X + i_{Y,Z}) + j_{X,Z})$$

then

$$\mathrm{assoc} : \quad ((X+Y)+Z) \to (X+(Y+Z)).$$

A similar construction applies to associativity arrows for products.

DEFINITION 3.4: Let Σ be an alphabet and **G** a distributive graph. The *text of an* (*imperative*) *program* of IMP(**G**) is a functor $\Gamma : \Sigma^* \to \mathrm{Expr}(\mathbf{G})$.

Therefore, the text of an imperative program Γ written in IMP(**G**) is a family of names of actions ($\Gamma_a : X \to X$, $a \in \Sigma$), where X is an expression of objects and Γ_a is a path in the expressions of arrows; X is called the name of the *state space* of the program. When Σ has just one letter in it, Γ is called the text of an *isolated program.*

3.3. The operation of a program

DEFINITION 3.5: The *length* of an expression, U, of objects of a distributive graph **G**, is the number of objects that appear in the expression counting O, I and repetitions.

For example, if the expression $U=(((A+O)+C)\times((C+A)\times I))$, then the length of U is 6.

Notation 3.1: (i) Denote the length of U by $|U|$.

(ii) If D is a set, then D^n is used to denote the set of all words of length n in the elements of D, and D^* is used to denote the set of all words in the elements of D.

(iii) Let e^n denote the word $ee \ldots e$ of length n in e.

Let **G** be a distributive graph. Suppose Φ is an assignment of a set $\Phi(A)$ to every object A of **G**, and let D be the disjoint union of the $\Phi(A)$'s, together with the elements e and $*$.

DEFINITION 3.6: Given an assignment Φ on the objects of **G**, we extend Φ to assign sets to expressions of objects of **G**, in such a way that if U is an expression of objects then $\Phi(U)$ is a set whose elements are words of length $|U|$ in D^*.

(i) $\Phi(I)=\{*\}$ and $\Phi(O)=\varnothing$.

(ii) If U and V are expressions of objects with $\Phi(U)$ and $\Phi(V)$ the assigned sets to U and V then:

$$\Phi((U\times V))=\{uv : u\in\Phi(U),\ v\in\Phi(V)\},$$
$$\Phi((U+V))=\{ue^{|V|} : u\in\Phi(U)\}\cup\{e^{|U|}v : v\in\Phi(V)\}.$$

Note 3.1: (i) If $u\in\Phi(U)$, then $u=u_1 u_2 \ldots u_{|U|}$ where $u_i\in\Phi(U_i)\cup\{e\}$ and U_i is I or an object of **G**.

(ii) Let $D=\sum_{A\in\mathcal{O}}\Phi(A)+\Phi(I)+\{e\}$. If U is an expression of objects, then $\Phi(U)\subset D^{|U|}\subset D^*$.

(iii) The functions:

$$\theta: \quad \Phi(U \times V) \to \Phi(U) \times \Phi(V)$$
$$uv \mapsto (u, v),$$
$$\varphi: \quad \Phi(U+V) \to \Phi(U) + \Phi(V)$$
$$ue^{|V|} \mapsto (u, 0)$$
$$e^{|U|}v \mapsto (u, 1)$$

are isomorphisms of sets. Therefore Φ assigns products in **Sets** to formal products and sums in **Sets** to formal sums.

DEFINITION 3.7: Let Φ be an assignment of sets to expressions of objects of a distributive graph **G** be as above. Suppose also that for every arrow $f: U \to V$ of **G**, there is an assigned set function $\Phi(f): \Phi(U) \to \Phi(V)$. We extend Φ to expressions of arrows of **G** as follows:

(i) Let X, Y, Z be expressions of objects with $\Phi(X)$, $\Phi(Y)$, $\Phi(Z)$ their assigned sets, then take the following assignments of functions to expressions of arrows of **G**.

$$\Phi(1_X): \quad \Phi(X) \to \Phi(X)$$
$$x \mapsto x$$
$$\Phi(p_{X,Y}): \quad \Phi((X \times Y)) \to \Phi(X)$$
$$xy \mapsto x$$
$$\Phi(q_{X,Y}): \quad \Phi((X \times Y)) \to \Phi(Y)$$
$$xy \mapsto y$$
$$\Phi(i_{X,Y}): \quad \Phi(X) \to \Phi((X+Y))$$
$$x \mapsto xe^{|Y|}$$
$$\Phi(j_{X,Y}): \quad \Phi(Y) \to \Phi((X+Y))$$
$$y \mapsto e^{|X|}y$$
$$\Phi(\delta^{-1}_{X,Y,Z}): \quad \Phi((X \times (Y+Z))) \to \Phi(((X \times Y) + (X \times Z)))$$
$$xye^{|Z|} \mapsto xye^{|X|+|Z|}$$
$$xe^{|Y|}z \mapsto e^{|X|+|Y|}xz$$
$$\Phi(!_X): \quad \Phi(X) \to \Phi(I)$$
$$x \mapsto *$$

$\Phi(¡_X)$: $\Phi(O) \to \Phi(X)$ is the unique arrow in **Sets** from $\varnothing$ to $\Phi(X)$

$$\Phi(\Delta_X):\quad \Phi(X)\to\Phi((X\times X))$$
$$x\mapsto xx$$
$$\Phi(\nabla_X):\quad \Phi((X+X))\to\Phi(X)$$
$$xe^{|X|}\mapsto x$$
$$e^{|X|}x\mapsto x$$

(ii) If $\alpha: U\to V$ and $\beta: X\to Y$ are expressions of arrows, with $\Phi(\alpha):\Phi(U)\to\Phi(V)$ and $\Phi(\beta):\Phi(X)\to\Phi(Y)$ their corresponding assigned set functions then the functions

$$\Phi((\alpha\times\beta)):\quad \Phi((U\times X))\to\Phi((V\times Y)$$
$$ux\mapsto\Phi(\alpha)(u)\,\Phi(\beta)(x)$$
$$\Phi((\alpha+\beta)):\quad \Phi((U+X))\to\Phi((V+Y))$$
$$ue^{|X|}\mapsto\Phi(\alpha)(u)\,e^{|Y|}$$
$$e^{|U|}x\mapsto e^{|V|}\,\Phi(\beta)(x).$$

(iii) If $\alpha_i: X_i\to X_{i+1}$ for $i=1,2,\ldots,n$ are expressions of arrows, then

$$\Phi(\alpha_n\circ\alpha_{n-1}\circ\ldots\circ\alpha_1)=\Phi(\alpha_n)\circ\Phi(\alpha_{n-1})\circ\ldots\circ\Phi(\alpha_1):\quad \Phi(X_1)\to\Phi(X_{n+1}).$$

Remark 3.2: (i) Let $fn\,\mathrm{sym}=\mathscr{A}\cup\{1, p, q, i, j, \delta^{-1}, ¡, !, \Delta, \nabla\}$. Notice that in the definition of $\Phi(c_X)$, $\Phi(c_{X,Y})$, $\Phi(c_{X,Y,Z})$ where $c\in fn\,\mathrm{sym}-\mathscr{A}$, we only need to know the lengths of X, Y, Z. In future, when the lengths are known, we only write $\Phi(c)$. The elements c in $fn\,\mathrm{sym}$ will be referred to as function symbols.

(ii) It is clear that the following are natural isomorphisms

$$\Phi(\alpha\times\beta)\cong\Phi(\alpha)\times\Phi(\beta),\qquad \Phi(\alpha+\beta)\cong\Phi(\alpha)+\Phi(\beta)$$
$$\Phi(1_X)\cong 1_{\Phi(X)},\qquad \Phi(\delta^{-1}_{X,Y,Z})\cong\delta^{-1}_{\Phi(X),\Phi(Y),\Phi(Z)}$$
$$\Phi(p_{X,Y})\cong p_{\Phi(X),\Phi(Y)},\qquad \Phi(i_{X,Y})\cong i_{\Phi(X),\Phi(Y)}$$
$$\Phi(q_{X,Y})\cong q_{\Phi(X),\Phi(Y)},\qquad \Phi(j_{X,Y})\cong j_{\Phi(X),\Phi(Y)}$$
$$\Phi(!_X)\cong !_{\Phi(X)},\qquad \Phi(¡_X)\cong(\Phi(¡_X))$$
$$\Phi(\Delta_X)\cong\Delta_{\Phi(X)},\qquad \Phi(\nabla_X)\cong\nabla_{\Phi(X)}$$

(iii) The assignment Φ takes associativity arrows of sums and of products to actual *identities* in **Sets**.

3.4. Elementary expressions

DEFINITION 3.8: An *elementary expression* is an expression of arrows defined as follows:

(i) All arrows of **G** are elementary expressions.

(ii) All expressions of arrows of the form c_X, $c_{X,Y}$, $c_{X,Y,Z}$, where $c \in fn\,\mathrm{sym} - \mathscr{A}$, are elementary expressions.

(iii) If $\zeta : U \to V$ is an elementary expression and $* \in \{+, \times\}$, then

(*a*) $(\zeta * 1_K) : (U * K) \to (V * K)$ and

(*b*) $(1_K * \zeta) : (K * U) \to (K * V)$ are elementary expressions.

Remark 3.3: Elementary expressions are (particular) expressions of arrows with at most one of the symbols in the set $fn\,\mathrm{sym} - \{1\}$, and do not include the composition symbol $\circ$.

PROPOSITION 3.1: *If* $\alpha : X \to Y$ *is an arrow in* Expr(**G**), *then there exists elementary expressions* $\xi_1, \xi_2, \ldots, \xi_n$ *such that* $\xi_n \circ \xi_{n-1} \circ \ldots \circ \xi_1 : X \to Y$ *and*

$$\Phi(\alpha) = \Phi(\xi_n \circ \xi_{n-1} \circ \ldots \circ \xi_1) : \quad \Phi(X) \to \Phi(Y).$$

Proof: The proof is by induction using the definition of expressions of arrows. If α is an arrow of **G** or is of the form c_X, $c_{X,Y}$, $c_{X,Y,Z}$, where $c \in fn\,\mathrm{sym}$, then it is clearly an elementary expression. Suppose the result is true for all elementary expressions smaller than α, where α is an expression of arrows with more than one function symbol. Then, by the definition of expressions of arrows, $\alpha = (\beta * \gamma)$, if $* \in \{\times, +\}$ or $\alpha = \alpha_n \circ \alpha_{n-1} \circ \ldots \circ \alpha_1$, where β, γ, $\alpha_1, \ldots, \alpha_n$ are expressions of arrows that are smaller than α, hence they can each be written as a composition of elementary expressions.

If $\alpha = \alpha_n \circ \alpha_{n-1} \circ \ldots \circ \alpha_1$, then the result follows immediately. Suppose $\alpha = (\beta * \gamma)$, then by the inductive hypothesis there exists elementary expressions

$$\mu_1 : \quad U_1 \to U_2, \qquad \mu_2 : \quad U_2 \to U_3, \ldots, \qquad \mu_s : \quad U_s \to U_{s+1}$$

and

$$\nu_1 : \quad V_1 \to V_2, \qquad \nu_2 : \quad V_2 \to V_3, \ldots, \qquad \nu_t : \quad V_t \to V_{t+1},$$

such that

$$\Phi(\beta) = \Phi(\mu_s \circ \mu_{s-1} \circ \ldots \circ \mu_1),$$
$$\Phi(\gamma) = \Phi(\nu_t \circ \nu_{t-1} \circ \ldots \circ \nu_1).$$

Therefore

$$\Phi(\alpha)=\Phi((\mu_s * 1_{V_{t+1}}) \circ \ldots \circ (\mu_2 * 1_{V_{t+1}}) \circ (\mu_1 * 1_{V_{t+1}}) \circ (1_{U_1} * \nu_t) \circ \ldots \circ (1_{U_1} * \nu_2) \circ (1_{U_1} * \nu_1))$$

is in the required form since the expressions $(\mu_i * 1_{V_t})$ and $(1_{U_1} * \nu_j)$ are elementary expressions by definition. □

COROLLARY 3.1: *For any program, P, there is a program which is a composition of elementary expressions whose behaviour is the same as P.*

3.5. Tokens

DEFINITION 3.9: Let $\mathscr{E}$ be the set of all elementary expressions constructed from **G**. We define a function

$$\text{tok}: \quad \mathscr{E} \to fn\,\text{sym} \times \mathbb{N}^6$$

on the elementary expressions as follows:

(i)

$\text{tok}(f)=(f, |X|, 0, 0, |U|, 0, 0)$ if $f: X \to U$ is an arrow of **G**

$\text{tok}(1_x)=(1, |X|, 0, 0, |X|, 0, 0)$

$\text{tok}(c_X)=(c, |X|, 0, 0, |U|, 0, 0)$ if $c \in \{\Delta, \nabla, !, ¡\}$ and $c: X \to U$.

$\text{tok}(c_{X,Y})=(c, |X|, |Y|, 0, |U|, 0, 0)$ if $c \in \{p, q, i, j\}$ and U the codomain of c.

$\text{tok}(\delta^{-1}_{X,Y,Z})=(\delta^{-1}, |X|, |Y|, |Z|, |U|, 0, 0)$ if $U=((X \times Y)+(X \times Z))$.

(ii) $\text{tok}((1_X * 1_Y))=(1, |X|, 0, 0, |X|, 0, |Y|)$.

(iii) If $\zeta \neq 1_X$ is an elementary expression such that

$$\text{tok}(\zeta)=(c, n_1, n_2, n_3, m, l, r),$$

then:

(*a*) $\text{tok}((\zeta * 1_K))=(c, n_1, n_2, n_3, m, l, r+|K|)$, and

(*b*) $\text{tok}((1_K * \zeta))=(c, n_1, n_2, n_3, m, l+|K|, r)$.

Remark 3.4: If $\xi: W \to Z$ is an elementary expression, then $\text{tok}(\xi)$ will be referred to as the *token* of ξ.

PROPOSITION 3.2: *Let* $\xi: W \to Z$ *be an elementary expression such that* $\text{tok}(\xi)=(c, n_1, n_2, n_3, m, l, r)$, *and* $n=n_1+n_2+n_3$, *then*

(i) *If* $w \in \Phi(W)$ *then* w *is in the form* uxv *where* u *is a word of length* l, $x \in \Phi(X) \cup e^n$, *where* $|X| = n$, v *is a word of length* r, *and* $\Phi(c) : \Phi(X) \to \Phi(Y)$ *for some expression* Y *such that* $|Y| = m$.

(ii)

$$\Phi(\xi)(uxv) = \begin{cases} u\,\Phi(c)(x)\,v, & \textit{if} \quad x \neq e^n \\ ue^m v, & \textit{if} \quad x = e^n. \end{cases}$$

Remark 3.5: As we have noted earlier, $\Phi(c)$ depends only on lengths in the domain of c; therefore, $\Phi(\xi)$ is string manipulation based on the knowledge of $\mathrm{tok}(\xi)$.

Proof: (i) This follows from the definition of $\mathrm{tok}(\xi)$ and elementary expressions.

(ii) We prove this by induction on the definition of elementary expressions. If $\xi \in \mathrm{char}$, then $r = l = 0$ in $\mathrm{tok}(\xi)$ and the result is immediate. Suppose r or $l > 0$ in $\mathrm{tok}(\xi)$ and the result is true for all expressions smaller than ξ. By definition of elementary expressions, $\xi = (\zeta * 1_K)$ or $\xi = (1_K * \zeta)$ for some expression K, and elementary expression $\zeta : W' \to Z'$ smaller than ξ.

If $\xi = (\zeta \times 1_K) : W' \times K \to W' \times K$, let $w' \in \Phi(W')$, $k \in \Phi(K)$, then

$$\begin{aligned} \Phi(\xi)(w'k) &= \Phi(\xi)(uxvk), \quad \text{by part (i)} \\ &= \Phi(\zeta)(uvx)\,\Phi(1_K)(k), \quad \text{by definition of } \Phi \\ &= \Phi(\zeta)(uvx)\,k, \quad \text{by definition of } \Phi \\ &= \begin{cases} u\,\Phi(c)(x)\,vk, & \text{if} \quad x \neq e^n \\ ue^m vk, & \text{if} \quad x = e^n \text{ by inductive hypothesis.} \end{cases} \end{aligned}$$

If $\xi = (\zeta + 1_K) : W' + K \to W' + K$, then

$$\begin{aligned} \Phi(\xi)(w' e^{|K|}) &= \Phi(\xi)(uxve^{|K|}), \quad \text{by part (i)} \\ &= \Phi(\zeta)(uxv)\,e^{|K|}, \quad \text{by definition of } \Phi \\ &= \begin{cases} u\,\Phi(c)(x)\,ve^{|K|}, & \text{if} \quad x \neq e^n \\ ue^m ve^{|K|}, & \text{if} \quad x = e^n \text{ by inductive hypothesis.} \end{cases} \end{aligned}$$

$\Phi(\xi)(e^l e^n e^r k) = e^l e^m e^r k$, by definition of Φ. $\square$

4. A UNIVERSAL IMP (G) PROGRAM

In this section we describe a universal IMP(G) program, $\mathcal{U}$, in IMP($\hat{\mathbf{G}}$) where $\hat{\mathbf{G}}$ is the graph $\mathbf{G}$ extended by data type $\mathbb{N}$, S_{data}, S_{char}, S_{tok} where $\Phi(\text{data})=D$, $\text{tok}=fn\,\text{sym}\times\mathbb{N}^6$ and $\Phi(\text{char})$ is the set with elements in:

$$\mathcal{O}\cup fn\,\text{sym}\cup\{,\ (\quad)\times+\circ\}$$

As we have discussed in the introduction, U is a program with state space of the form

$$X_{\mathcal{U}}=S_{\text{char}}\times S_{\text{data}}+Z.$$

The state space of any IMP(G) program P is an expression in the objects of $\mathbf{G}$, so any state of P is a word in the elements of these objects plus $*$ and e. Therefore, any state of P is an element in D^*. It then follows that any state of P may be represented as an element of S_{data}. Similarly, the text of the program P, is a word in the characters, and hence may be represented as an element of S_{char}.

Note 4.1. We are assuming that the program is written as a composite of elementary arrows.

The action $\text{act}_{\mathcal{U}}:X_{\mathcal{U}}\to X_{\mathcal{U}}$, which we will describe precisely below, will have the property that; if (t, x_0) is the initial state of $\mathcal{U}$ with t the text of the program P, then the sequence of global states of $\mathcal{U}$ is

$$(t, x_0),\quad (t, x_1),\quad (t, x_2),\quad \ldots$$

where $x_0, x_1, x_2, \ldots$ is the behaviour of P with initial state x_0.

The way we shall produce act_U is by constructing a pseudofunction

$$\begin{aligned}\varphi_1:\quad & S_{\text{char}}\times S_{\text{data}}\nrightarrow S_{\text{char}}\times S_{\text{data}}\\ & (t, x)\mapsto(t, x')\end{aligned}$$

such that if t is the text of the program P and x is a state of P, then $x'=\text{act}_P(x)$. If $\text{act}_{\mathcal{U}}=\text{iter}(\varphi_1)$, as in proposition 2.6, then $\text{act}_{\mathcal{U}}$ is the required program.

The pseudofunction $\varphi_1=(\Delta_{S_{\text{char}}}\wedge 1_{S_{\text{data}}});\ (1_{S_{\text{char}}}\wedge\varphi_2\wedge 1_{S_{\text{data}}});\ (1_{S_{\text{char}}}\wedge\varphi_3)$, where the pseudofunction

$$\varphi_2:\quad S_{\text{char}}\nrightarrow S_{\text{tok}}$$

translates a program

$$\xi_n \circ \xi_{n-1} \circ \ldots \circ \xi_1,$$

(with $\xi_1, \xi_2, \ldots, \xi_n$ elementary expressions) to the stack of tokens:

$$\mathrm{tok}(\xi_n)\,\mathrm{tok}(\xi_{n-1}) \ldots \mathrm{tok}(\xi_1).$$

and the pseudofunction

$$\varphi_3: \quad S_{\mathrm{tok}} \times S_{\mathrm{data}} \leftrightarrow S_{\mathrm{data}}$$

carries out the effect of the string of tokens in S_{tok} on the data in S_{data}. It is constructed, as in proposition 2.5, from another pseudofunction

$$\varphi_4: \quad S_{\mathrm{tok}} \times S_{\mathrm{data}} \leftrightarrow S_{\mathrm{tok}} \times S_{\mathrm{data}} + S_{\mathrm{data}}$$

where the order $\mathrm{ord}: S_{\mathrm{tok}} \times S_{\mathrm{data}} \to \mathbb{N}$ is the size of the stack of tokens.

The pseudofunction $\varphi_4 = (\mathrm{pop} \wedge 1_{S_{\mathrm{data}}});\ ((1_{S_{\mathrm{tok}}} \wedge \varphi_5) \vee 1_{S_{\mathrm{data}}})$, where δ is the isomorphism $\delta: (S_{\mathrm{tok}} \times \mathrm{tok} + I) \times S_{\mathrm{data}} \to S_{\mathrm{tok}} \times \mathrm{tok} \times S_{\mathrm{data}} + S_{\mathrm{data}}$ and

$$\varphi_5: \quad \mathrm{tok} \times S_{\mathrm{data}} \leftrightarrow S_{\mathrm{data}}$$

carries out of the effect of one token on data in S_{data}. It is clear that the effect of φ_3 is to repeatedly carry out φ_5 until the stack of tokens is empty.

To define φ_5 notice that $\mathrm{toc} = fn\,\mathrm{sym} \times \mathbb{N}^6 \cong n \,.\, \mathbb{N}^6$ where n is the number of function symbols, then by corollary 2.1, it suffices to give n pseudofunctions:

$$\varphi_c: \quad \mathbb{N}^6 \times S_{\mathrm{data}} \leftrightarrow S_{\mathrm{data}}, \qquad (c \in fn\,\mathrm{sym})$$

As we have seen in the last section, each of the pseudofunctions, φ_c, is a simple string manipulation. By the above discussion we have reduced the description of $\mathscr{U}$ to the pseudofunctions:

$$\varphi_2: \quad S_{\mathrm{char}} \leftrightarrow S_{\mathrm{tok}}$$
$$\varphi_c: \quad \mathbb{N}^6 \times S_{\mathrm{data}} \leftrightarrow S_{\mathrm{data}}, \qquad (c \in fn\,\mathrm{sym}).$$

We will describe just one of these, namely φ_Δ. Two additional stacks of type *data* will be used and for simplicity of notation we write S for the stack we start with. The others will be referred to as S_1, and S_2. They will be initialised by the pseudofunction $\mathrm{init}(S) = \mathrm{push}: I \leftrightarrow S$. The pseudofunction $\varphi_\Delta = \rho_1;\ \rho_2;\ \ldots;\ \rho_6$. The pseudofunction ρ_1 projects of unnecessary numbers in $\mathbb{N}^6$, and is given as:

$$\rho_1=(p \wedge 1_S): \quad \mathbb{N}^6 \times S \leftrightarrow \mathbb{N}^2 \times S$$

where $p: \mathbb{N}^6 \to \mathbb{N}^2$ is the projection given by $p(n_1, n_2, n_3, m, l, r)=(n_1, r)$. The pseudofunction ρ_2 initializes S_1 and S_2, it is given as:

$$\rho_2=\theta; \quad (1_{N^2S} \wedge \text{init}(S_1) \wedge \text{init}(S_2)): \quad \mathbb{N}^2 \times S \leftrightarrow \mathbb{N}^2 \times S \times S_1 \times S_2$$

where θ is the isomorphism $\mathbb{N}^2 \times S \cong \mathbb{N}^2 \times S \times I \times I$. The pseudofunctions $\rho_3, \rho_4, \rho_5, \rho_6$ are:

$$\rho_3: \quad \mathbb{N}^2 \times S \times S_1 \times S_2 \leftrightarrow \mathbb{N} \times S \times S_1 \times S_2$$

$$\rho_4: \quad \mathbb{N} \times S \times S_1 \times S_2 \leftrightarrow S \times S_1 \times S_2$$

$$\rho_5: \quad S \times S_1 \times S_2 \leftrightarrow S \times S_1$$

$$\rho_6: \quad S \times S_1 \leftrightarrow S$$

where ρ_3 pushes the top r cells (r in the second component of $\mathbb{N}^2$) of S onto S_1, ρ_4 pushes the top $n \in \mathbb{N}$ cells of S onto S_1 and onto S_2, ρ_5 pushes S_2 on top of S, and ρ_6 pushes S_1 on top of S.

The pseudofunction ρ_3 is constructed, as in proposition 2.5, from another pseudofunction $\rho_7=\theta_1; ((1_{N^2} \wedge \theta_2) \vee 1_{NSS_1S_2})$ given by:

$$\rho_7: \quad \mathbb{N}^2 \times S \times S_1 \times S_2 \leftrightarrow \mathbb{N}^2 \times S \times S_1 \times S_2 + \mathbb{N} \times S \times S_1 \times S_2$$

$$(n_1, r, x_1, x_2 \ldots x_p, s_1, s_2)$$

$$\mapsto (n_1, r-1, x_1 x_2 \ldots x_{p-1}, s_1 x_p, s_2), \qquad \text{if} \quad r \neq 0, \quad p \geqq 1$$

$$(n_1, r, s, s_1, s_2) \mapsto \begin{cases} (n_1, r-1, s, s_1, s_2) & \text{if} \quad r \neq 0, \quad s=o \\ (n_1, s, s_1, s_2) & \text{if} \quad r=0 \end{cases}$$

where the function ord is the projection onto the second component in $\mathbb{N}^2$. The pseudofunction $\theta_1=(1_N \wedge \mathit{predecessor} \wedge 1_{SS_1S_2})$; δ_1, where is the isomorphism

$$\delta_1: \quad \mathbb{N} \times (\mathbb{N}+I) \times S \times S_1 \times S_2 \leftrightarrow \mathbb{N}^2 \times S \times S_1 \times S_2 + \mathbb{N} \times S \times S_1 \times S_2$$

and

$$\theta_2: \quad S \times S_1 \times S_2 \leftrightarrow S \times S_1 \times S_2.$$

where $\theta_2=(\text{pop} \wedge 1_{S_1, S_2})$; δ_2; $(1_S \wedge \text{push} \wedge 1_{S_2}) \vee (\text{push} \wedge 1_{S_1S_2})$; $\nabla_{SS_1S_2}$, where δ_2 is a distributive law isomorphism. So the effect of ρ_7 is to first test if n in the second component of $\mathbb{N}^2$ is 0; via θ_1, if $n \neq 0$ then the top cell on S is pushed on top of S_1, via θ_2. If $n=0$, ρ_7 idles.

The pseudofunction ρ_4 is constructed, as in proposition 2.5, from another pseudofunction

$$\rho_8: \quad \mathbb{N} \times S \times S_1 \times S_2 \nrightarrow \mathbb{N} \times S \times S_1 \times S_2 + S \times S_1 \times S_2$$

where the function ord is the first projection, ord : $\mathbb{N} \times S \times S_1 \times S_2 \to \mathbb{N}$, therefore, it gives the size of $\mathbb{N}$.

The pseudofunction ρ_5 is constructed, as in proposition 2.5, from another pseudofunction

$$\rho_9: \quad S \times S_1 \times S_2 \nrightarrow S \times S_1 \times S_2 + S \times S_1$$

where the function ord : $S \times S_1 \times S_2 \to \mathbb{N}$ is the size of S_2.

The pseudofunction ρ_6 is constructed, as in proposition 2.5, from another pseudofunction

$$\rho_{10}: \quad S \times S_1 \nrightarrow S \times S_1 + S$$

where the function ord : $S \times S_1 \to \mathbb{N}$ is the size of S_1. The pseudofunctions ρ_8, ρ_9, and ρ_{10} are constructed in a similar way to ρ_7.

REFERENCES

1. R. F. C. Walters, An Imperative Language based on Distributive Categories, in *Mathematical Structures for Computer Science* (to appear).
2. R. F. C. Walters, *Categories and Computer Science*, Carslaw Publications, 1991, Cambridge University Press, 1992.
3. R. F. C. Walters, *Lecture at the Conference on Categories and Computer Science*, Durham, July 1991.