Polynomial Relators
Extended Abstract

Roland C. Backhouse
Eindhoven University of Technology, 5600 MB Eindhoven, NL

Peter J. de Bruin
Rijksuniversiteit Groningen, 9700 AV Groningen, NL

Paul Hoogendijk
Eindhoven University of Technology, 5600 MB Eindhoven, NL

Grant Malcolm
Oxford University, Oxford OX1 3QD, UK

Ed Voermans
Eindhoven University of Technology, 5600 MB Eindhoven, NL

Jaap van der Woude
CWI, 1009 AB Amsterdam, NL
Eindhoven University of Technology, 5600 MB Eindhoven, NL

Abstract

This paper reports ongoing research into a theory of datatypes based on the calculus of relations. A fundamental concept introduced here is the notion of "relator" which is an adaption of the categorical notion of functor. Axiomatisations of polynomial relators (that is relators built from the unit type and the disjoint sum and cartesian product relators) are given and their properties extensively catalogued. The effectiveness of the calculus is illustrated by the construction of several natural isomorphisms and natural simulations between relators.

1 Introduction

Research is underway into a theory of datatypes based on the calculus of relations. A fundamental concept within the theory is that of "relator" which takes the place of the notion of "functor" in a category-theory inspired development of functional programming.

In order that our theory be of any use we need to ensure that we can indeed define some non-trivial relators. It is well accepted that any useful theory of types should contain as a bare minimum the three components: a unit type, a disjoint sum operator and a cartesian product operator. In this paper we axiomatise these three within the calculus of relations and catalogue their properties. Following established terminology (see for example Manes and Arbib [6]) we call the relators so obtained the polynomial relators.

The main goal of our endeavour is to develop a calculus admitting compact and clear derivation of programs. To this end emphasis is placed on the identification of morphisms associated with each relator and their so-called "fusion" and "naturality" properties. Fusion properties characterise the circumstances under which a composition of relations, one of which is a morphism, can be combined into just one morphism (or vice-versa when a morphism can be "defused" into a composition of relations one of which remains a morphism). Naturality properties offer an alternative way of expressing monotonicity and fusion properties of morphisms; from a programming point of view their significance emerges when one tackles the problem of constructing simulations of one relator by another or isomorphisms between relators. The effectiveness of the calculus is demonstrated by exhibiting one such construction.

Because of space limitations almost all proofs have been omitted from this extended abstract as well as all discussion of continuity properties. The complete paper is available on request from the authors. A companion paper [3] provides more motivation for the development of the theory and takes it further into the domain of inductively-defined types.

2 The Algebraic Framework

In this section we summarise relational algebra. For pedagogic reasons we decompose the algebra into three layers with interfaces between the layers plus two special axioms, one discussed here and one in a later section. The algebra is standard within the bounds of "scientific freedom".

2.1 Plat Calculus

Let $\mathcal{A}$ be a set, the elements of which are to be called *specs*. On $\mathcal{A}$ we impose the structure of a complete, completely distributive, complemented lattice, $(\mathcal{A}, \sqcap, \sqcup, \neg, \top\!\top, \bot\!\bot)$, where "$\sqcap$" and "$\sqcup$" are associative and idempotent, binary infix operators with unit elements "$\top\!\top$" and "$\bot\!\bot$", respectively, and "$\neg$" is the unary prefix operator denoting complement (or negation). We call such a structure a *plat*, the "p" standing for power set and "lat" standing for lattice. Since the structure is so well known and well documented we shall assume a high degree of familiarity with it.

2.2 Composition

The second layer is the monoid structure for composition, $(\mathcal{A}, \circ, I)$, where $\circ$ is an associative binary infix operator with unit element I. The interface between these two layers is: $\circ$ is coordinatewise universally "cup-junctive". I.e. for $V, W \subseteq \mathcal{A}$, $(\sqcup V) \circ (\sqcup W) \;=\; \sqcup(P, Q :\; P \in V \;\wedge\; Q \in W :\; P \circ Q)$.

2.3 Reverse

The third layer is the "reverse structure", $(\mathcal{A}, \cup)$, where "$\cup$" is a unary postfix operator such that it is its own inverse. The interface with the first layer is that "$\cup$" is an isomorphism of plats. I.e. for all $P, Q \in \mathcal{A}$, $P \sqsupseteq Q \;\equiv\; P\cup \sqsupseteq Q\cup$.

The interface with the second layer is that "$\cup$" is an isomorphism between the two monoid structures $(\mathcal{A}, \;;, I)$ and $(\mathcal{A}, \circ, I)$ where $R \;;\; S \;=\; S \circ R$.

2.4 The Middle Exchange Rule

To the above axioms we now add an axiom that acts as an interface between all three layers.

$$\neg Y \;\sqsupseteq\; P \circ \neg X \circ Q \;\equiv\; X \;\sqsupseteq\; P\cup \circ Y \circ Q\cup$$

The middle exchange rule is a variant of the so-called Schröder rule. (See, for example, [10] for historical references.)

3 Foundations

The purpose of this section is to build up a vocabulary for our later discussion of the properties of morphisms. In order to avoid possible confusion with existing terminology we make a complete reappraisal of what is meant by "type", "function", "type constructor" etc. Nevertheless, it should be emphasised that — with the important exception of the notion of "relator" — *the concepts defined in this section, and their properties, are amply documented in the mathematical literature and we make no claim to originality.*

3.1 Monotypes

We say that spec A is a *monotype* iff $I \sqsupseteq A$. Note that for monotypes A and B

$$A \;=\; I \sqcap A \;=\; A\cup \;=\; A \circ A \tag{1}$$
$$A \circ B \;=\; B \circ A \;=\; A \sqcap B \tag{2}$$

We need to refer to the "domain" and "co-domain" (or "range") of a spec. In order to avoid unhelpful operational interpretations we use the terms *left-domain* and *right-domain* instead. These are denoted by "<" and ">", respectively, and defined by

$$R< \;\widehat{=}\; I \sqcap (R \circ R\cup) \quad \text{and} \quad R> \;\widehat{=}\; I \sqcap (R\cup \circ R) \tag{3}$$

Since all the operators involved in their definition are monotonic it follows that "<" and ">" are monotonic. Relational calculus yields the following alternative definitions defining $R<$ and $R>$ as the smallest monotypes satisfying the equations in A, $A \circ R = R$ and $R \circ A = R$, respectively.

$$\forall(A : I \sqsupseteq A : A \circ R = R \;\equiv\; A \sqsupseteq R<) \tag{4}$$
$$\forall(A : I \sqsupseteq A : R \circ A = R \;\equiv\; A \sqsupseteq R>) \tag{5}$$

We sometimes write $R \in S{\sim}T$ as a synonym for $S \circ R \;=\; R \;=\; R \circ T$

It is immediate from (4) and (5) that

$$R< \circ R \;=\; R \;=\; R \circ R> \tag{6}$$

3.2 Imps and Co-imps

In this subsection we define "imps" and "co-imps" as special classes of specs. In the relational model an "imp" is a function.

Definition 7

(a) A spec f is said to be an *imp* if and only if $I \sqsupseteq f \circ f\cup$.
(b) A spec f is said to be a *co-imp* if and only if $f\cup$ is an imp.
(c) A spec is said to be a *bijection* if and only if it is both an imp and a co-imp.
□

We shall say that f is a bijection *between A and B* if it is a bijection and $f< = A$ and $f> = B$. Note that if this is the case then both A and B are monotypes and $A = f \circ f\cup$ and $B = f\cup \circ f$. The notation "$A \cong B$" (read as A is *isomorphic* to B) signifies the existence of a bijection between A and B.

Theorem 8 Composition preserves imps, co-imps and bijections.
□

The intended interpretation is that an "imp" is an "imp"lementation. On the other hand, it is not the intention that all implementations are "imps". Apart from their interpretation imps have an important distributive property not enjoyed by arbitrary specs, namely:

Theorem 9 If f is an imp and g a co-imp then, for all specs R and S,

(a) $(R \sqcap S) \circ f = (R \circ f) \sqcap (S \circ f)$
(b) $g \circ (R \sqcap S) = (g \circ R) \sqcap (g \circ S)$
□

Monotypes are examples of bijections. More generally, the requirement of being a function is the requirement of being single-valued on the "domain" of the function. The domain and range are made explicit in the following.

Definition 10 For monotypes A and B we define the set $A \longleftarrow B$ by $f \in A \longleftarrow B$ whenever $I \sqsupseteq f \circ f\cup \wedge f> = B$. The nomenclature "$f \in A \longleftarrow B$" is verbalised by saying that "*f is an imp to A from B*".
□

We should stress that the two set-forming operations "$\sim$" and "$\longleftarrow$" do *not* form an essential part of our theory but are included in order that the reader may relate their existing knowledge of type structures to the present theory. In the sequel we shall often state properties of the domain-forming operations "<" and ">" and immediately transcribe them into properties of "$\sim$" and/or "$\longleftarrow$". We prefer the statements about the domains for two reasons: they offer a better separation of concerns and are thus calculationally more useful, and they can be stated with fewer dummies (and indeed in some cases with no dummies, although we don't go that far).

To avoid repeating assumptions and to assist the reader's understanding we continue to use the conventions that capital letters $A, B, C, \ldots$ at the beginning

of the alphabet denote monotypes, small letters $f, g, h, \ldots$ denote imps or co-imps, and capital letters $R, S, T, \ldots$ at the end of the alphabet denote arbitrary specs.

Finally, let us remark that the unconventional direction of the arrow in the statement "$f \in A \longleftarrow B$" is entirely dictated by the choice to denote function application with the function name to the left of its argument. (We owe the suggestion to deviate from convention to Meertens [7].)

3.3 Relators

In categorical approaches to type theory a parallel is drawn between the notion of type constructor and the categorical notion of "functor", thereby emphasising that a type constructor is not just a function from types to types but also comes equipped with a function that maps arrows to arrows. For an informative account of this parallel see, for example, [6]. In this subsection we propose a modest extension to the notion of functor to which we give the name "relator".

Definition 11 A *relator* is a function, F, from specs to specs such that

(a) $I \sqsupseteq F.I$ (b) $R \sqsupseteq S \Rightarrow F.R \sqsupseteq F.S$
(c) $F.(R \circ S) = F.R \circ F.S$ (d) $F.(R\cup) = (F.R)\cup$
□

In view of (11d) we take the liberty of writing simply "$F.R\cup$" without parentheses, thus avoiding explicit use of the property.

The above ostensibly defines a *unary* relator but we also wish to allow it to serve as the definition of a relator mapping an m-ary *vector* of specs into an n-ary *vector* of specs, for some natural numbers m and n. (This is necessary in order to allow the theory to encompass what are variously called "mutually recursive type definitions" and "many-sorted algebras".) The mechanism by which we can do this is to assume that all the constants appearing in the definition ("$=$", "$\sqsupseteq$", "I", "$\circ$" and "$\cup$") are silently "lifted" to operate on vectors. One case that we use in particular is when F maps a pair of specs into a spec. We refer to such relators as *binary* relators and choose to denote them by infix operators. Thus, if $\otimes$ denotes a binary relator, its defining properties would be spelt out as follows.

(a) $I \sqsupseteq I \otimes I$
(b) $R \sqsupseteq S \wedge U \sqsupseteq V \Rightarrow R \otimes U \sqsupseteq S \otimes V$
(c) $(R \circ S) \otimes (U \circ V) = (R \otimes U) \circ (S \otimes V)$
(d) $(R\cup) \otimes (S\cup) = (R \otimes S)\cup$

As already announced relators commute with the domain operators.

Theorem 12 If F is a relator then $F.(R>) = (F.R)>$ and $F.(R<) = (F.R)<$.
□

In view of theorem 12 we write "$F.R<$" and "$F.R>$" without parentheses, again in order to avoid explicit mention of the properties.

The following theorem allows a comparison to be made with our definition of "relator" and the definition of "functor" (in the category of sets).

Theorem 13 If F is a relator then

(a) A is a monotype $\Rightarrow$ $F.A$ is a monotype
(b) f is an imp $\Rightarrow$ $F.f$ is an imp
(c) f is a co-imp $\Rightarrow$ $F.f$ is a co-imp
(d) $f \in A \longleftarrow B$ $\Rightarrow$ $F.f \in F.A \longleftarrow F.B$
(e) $R \in A \sim B$ $\Rightarrow$ $F.R \in F.A \sim F.B$
□

4 Natural Polymorphism

Any discussion of a theory of datatypes would be incomplete without a discussion of polymorphism. This is particularly true here because our theory is principally a theory of two sets of polymorphic functions — the relators and their associated morphisms. Relators are polymorphic in the sense that they may be applied to arbitrary specs irrespective of the domains of the argument spec. Such a statement is, however, somewhat banal since it says nothing about the mathematical nature of the claimed polymorphism. In this section we shall argue that relators are "naturally polymorphic". The latter notion is an adaptation and extension of the notion of "natural transformation" in category theory; the definition that we use is based on the work of de Bruin [5] which work was anticipated by Reynolds [8].

4.1 Higher-Order Spec Algebras

Expressing the natural polymorphism of relators (and other functions or relations) requires the notion of higher-order spec algebra which we now define.

Let SPEC $= (\mathcal{A}, \sqsupseteq, I, \circ, \cup)$ be a spec-algebra. Then the algebra of binary relations on specs $\overline{\text{SPEC}}$ is defined to be $(\overline{\mathcal{A}}, \overline{\sqsupseteq}, \overline{I}, \overline{\circ}, \overline{\cup})$ where

$$\begin{aligned} \overline{\mathcal{A}} &= \mathsf{P}(\mathcal{A} \times \mathcal{A}) \\ \overline{\sqsupseteq} &= \supseteq \end{aligned}$$

and, using the notation $x \langle R \rangle y$ instead of $(x, y) \in R$,

$$\begin{aligned} x \langle \overline{I} \rangle y &\equiv x = y \\ x \langle R \mathbin{\overline{\circ}} S \rangle z &\equiv \exists (y :: x \langle R \rangle y \wedge y \langle S \rangle z) \\ x \langle R \overline{\cup} \rangle y &\equiv y \langle R \rangle x \end{aligned}$$

for all $R, S \in \overline{\mathcal{A}}$ and all x, y and $z \in \mathcal{A}$. With these definitions, $\overline{\text{SPEC}}$ is also a spec algebra. We call $\overline{\text{SPEC}}$ a *higher-order spec algebra.*

The imps of $\overline{\text{SPEC}}$ are (partial) functions to $\mathcal{A}$ from $\mathcal{A}$. Specifically, the function f from $\mathcal{A}$ to $\mathcal{A}$ is identifed with the relation f on $\mathcal{A} \times \mathcal{A}$ where

$$x \langle f \rangle y \equiv x = f.y$$

for all $x, y \in \mathcal{A}$.

The monotypes of $\overline{\text{SPEC}}$ can be identified with the subsets of $\mathcal{A}$. That is, a binary relation $\overline{A}$ in $\overline{\mathcal{A}}$ is a monotype if and only if there is an element A of $\mathsf{P}(\mathcal{A})$ such that

$$\forall(x, y :: \; x \,\langle \overline{A} \rangle\, y \;\equiv\; x = y \wedge x \in A) \qquad (14)$$

The operators "$\sim$" and "$\longleftarrow$" were defined in section 3.2 as set-forming operators. Using (14) to identify monotypes of $\overline{\text{SPEC}}$ with subsets of $\mathcal{A}$, we may identify "$\sim$" and "$\longleftarrow$" with elements of $\overline{\mathcal{A}}$, specifically with binary relations on elements of $\mathcal{A}$ that are subsets of the identity relation $\overline{I}$. To reinforce this identification we corrupt the normal usage of the belongs-to symbol "$\in$" by the following definition. For spec R and relation $\overline{S}$ we define

$$R \mathrel{\overline{\in}} \overline{S} \;\equiv\; R \,\langle \overline{S} \rangle\, R$$

Of course, $\overline{\text{SPEC}}$ can itself serve as the basis for the construction of a second algebra of binary relations $\overline{\overline{\text{SPEC}}}$, and in this way one can construct an infinite hierarchy of spec algebras. The relators and morphism constructors of one algebra are then total imps in the next higher order algebra. Maintaining the distinction between the levels has been one reason why we have continually distinguished between "specs" and "relations", and between "imps" and "functions".

In this section we define three more relations which we call the naturality operators. The operators will be used at various levels in the hierarchy of SPEC algebras but we do not bother to decorate the different uses with a bar to indicate the level of use. Similarly, we use the undecorated symbols "$\longleftarrow$", "$\sim$", "$\circ$", "$\cup$" etc. at all levels in the hierarchy.

4.2 The Naturality Operators

Definition 15 (The Naturality Operators) Let R and S be specs. Then we define the relations $R \dot{\leftsquigarrow} S$, $R \dot{\rightsquigarrow} S$ and $R \dot{\leftrightsquigarrow} S$ by

(a) $U \,\langle R \dot{\leftsquigarrow} S \rangle\, V \;\equiv\; R \circ V \sqsupseteq U \circ S$

(b) $U \,\langle R \dot{\rightsquigarrow} S \rangle\, V \;\equiv\; R \circ V \sqsubseteq U \circ S$

(c) $U \,\langle R \dot{\leftrightsquigarrow} S \rangle\, V \;\equiv\; R \circ V = U \circ S$

□

The above definition of the $\dot{\leftsquigarrow}$ operator was introduced in [1]; it is related by part (a) of the following theorem to definitions introduced variously by deBruin [5], Reynolds [8] and Wadler [11].

Theorem 16

(a) If R and S are relations and f and g are total functions then
$f \,\langle R \dot{\leftsquigarrow} S \rangle\, g \;\equiv\; \forall(u, v :: \; f.u \,\langle R \rangle\, g.v \Leftarrow u \,\langle S \rangle\, v)$

(b) If R and S are relations and $f\cup$ and $g\cup$ are total functions then
$f \,\langle R \dot{\rightsquigarrow} S \rangle\, g \;\equiv\; \forall(u, v :: \; u \,\langle R \rangle\, v \Rightarrow f\cup.u \,\langle S \rangle\, g\cup.v)$

(c) If R and S are relations and f and g are total, surjective bijections then
$f \,\langle R \dot{\leftrightsquigarrow} S \rangle\, g \;\equiv\; \forall(u, v :: \; f.u \,\langle R \rangle\, g.v \equiv u \,\langle S \rangle\, v)$

□

See [1] for the (straightforward) proof of part (a) of this theorem.

4.3 Naturality of Relators, Reverse and Composition

Three immediate examples of naturality properties are furnished by the relators, reverse and composition. For details of the latter two see the main paper. For relators we have:

Theorem 17 (Naturality of Relators) If F is a relator then for all specs R and S

(a) $F \in (F.R \mathrel{\dot{\sim}} F.S) \mathrel{\dot{\sim}} (R \mathrel{\dot{\sim}} S)$
(b) $F \in (F.R \mathrel{\dot{\rightarrow}} F.S) \mathrel{\dot{\sim}} (R \mathrel{\dot{\rightarrow}} S)$
(c) $F \in (F.R \mathrel{\dot{\leftrightarrow}} F.S) \mathrel{\dot{\sim}} (R \mathrel{\dot{\leftrightarrow}} S)$
□

Nowhere in this document do we hazard a definition of "natural polymorphism". Theorem 17 does, however, express precisely what we intend by the informal statement that relators are "naturally polymorphic". Similar theorems are proved later about the basic constituents of cartesian products and disjoint sums. In each case the theorem involves a universal quantification over specs, and it is in this sense that the spec in question is "polymorphic". The adjective "naturally" is added to suggest the link with "natural transformation" in category theory and to avoid confusion of our notion of polymorphism with existing notions.

5 The Unit Type

The unit type corresponds to a set with only one element; not a particularly interesting type, but nevertheless useful as a building block for constructing more complex data structures. The theory presented so far doesn't provide a vocabulary for talking about elements, only for talking about specs: this is not unintentional since a goal of our work has always been to minimise the incidence of point-wise arguments. In keeping with this goal, we adopt a rather abstract view of data types, and take a roundabout route to characterise the unit type.

5.1 The Cone Rule

We begin by postulating an axiom dubbed "the cone rule". This axiom could equally well have been included in section 2. It has been included here because it is only within the axiomatisation of the unit type that we make any use of the rule. Elsewhere (e.g. [10]) the cone rule is called "Tarski's rule."

The Cone Rule

$$\top \circ R \circ \top = \top \quad \equiv \quad R \neq \bot\!\!\!\bot$$

As a partial motivation for the cone rule we ask the reader to compare it with the following consequence of the RS rule.

Lemma 18 For all specs R, the following statements are all equivalent:

(a) $\perp\!\!\!\perp = R$
(b) $\perp\!\!\!\perp = R \circ \top\!\!\top$
(c) $\perp\!\!\!\perp = \top\!\!\top \circ R$
(d) $\perp\!\!\!\perp = \top\!\!\top \circ R \circ \top\!\!\top$
(e) $\perp\!\!\!\perp = R<$
(f) $\perp\!\!\!\perp = R>$

□

One consequence of the cone rule is that $\top\!\!\top$ and $\perp\!\!\!\perp$ are different. More significantly, by combining the cone rule and lemma 18, one sees that the spec $\top\!\!\top \circ R \circ \top\!\!\top$ is always either $\top\!\!\top$ or $\perp\!\!\!\perp$ whatever the value of spec R. We say that the function mapping R to $\top\!\!\top \circ R \circ \top\!\!\top$ is *boolean-valued*; the cone rule itself is an abstract and concise way of expressing the proposition that, considered as a set of pairs, spec R either contains no elements or contains at least one element.

Another consequence of the cone rule, that we mention for later use, is the following:

Lemma 19 $\quad \perp\!\!\!\perp = R \circ \top\!\!\top \circ S \quad \equiv \quad \perp\!\!\!\perp = R \vee \perp\!\!\!\perp = S$

□

5.2 The Axioms

In order to capture the notion of a unit type we need to express a sort of dual to the cone rule, namely that there is a non-empty spec which, when viewed as a set of pairs, consists of at most one pair the two components of which are identical. Specifically, we posit the existence of a spec, denoted **1**, such that

$$\perp\!\!\!\perp \neq \mathbf{1} \quad \text{and} \quad I \sqsupseteq \mathbf{1} \circ \top\!\!\top \circ \mathbf{1} \tag{20}$$

5.3 An Atomic Monotype

There are several ways to convince oneself that axiom (20) is indeed what we seek. One is to interpret the axioms in the relational model; another is to explore the consequences of the axioms within the theory itself. We would not discourage the reader from doing the former, but prefer ourself to emphasise the latter. We verify, first, that the unit type is an "atomic" monotype ("atomic" to be defined shortly) and, second, that it is a "terminal object" in the sense of category theory. Finally, we summarise certain basic properties of the unit type.

Theorem 21 **1** is a monotype.

□

We now define an *atom* to be a spec R such that $R \sqsupseteq X \Rightarrow \perp\!\!\!\perp = X \vee R = X$, for every spec X, Clearly, $\perp\!\!\!\perp$ is an atom. In general, the relational interpretation of an atom is a set of pairs containing at most one element.

Theorem 22 **1** is an atom.

□

Properties (20), (21) and (22) express, respectively, that **1** is non-empty, and is a monotype corresponding to a set containing at most one element.

5.4 Terminality

The abstractness in the definition of the unit type consists, in part, of the fact that the unit type characterises *any* one-element set (or, if you prefer, is modelled by any one-element set); the identity of the element is irrelevant. In category theory a unit type is characterised by the following so-called "terminality" property: for each set A, there is one, and only one, function — commonly denoted by $!_A$ — in $\mathbf{1} \longleftarrow A$. Letting $!$ denote $\mathbf{1} \circ \top\!\!\top$, this characterisation of the unit type is mimicked in our theory by the following two consequences of axiom (20). For all monotypes A,

$$! \circ A \in \mathbf{1} \longleftarrow A \text{ and } R \in \mathbf{1} \sim A \ \wedge \ R\rangle = A \ \equiv \ R = ! \circ A \quad (23)$$

Thus the categorical function $!_A$ is rendered by the imp $! \circ A$.

Equivalent, more succinct, and more fundamental, renderings of (23) are

$$! \text{ is an imp, and } \mathbf{1} = \mathbf{1} \circ \top\!\!\top \circ \mathbf{1} \quad (24)$$

from which follows

$$\mathbf{1} \sqsupseteq R\langle \ \equiv \ R = ! \circ R\rangle \quad (25)$$

It is also clear from these properties that $\mathbf{1}$ is unique up to isomorphism: if $\mathbf{1}'$ is also a unit type then $\mathbf{1} \circ \top\!\!\top \circ \mathbf{1}'$ is a bijection between $\mathbf{1}$ and $\mathbf{1}'$.

5.5 A Summary of Basic Properties

The "foundations" that were laid in sections 3 and 4 were not without purpose. In this and later sections we shall continually ask a number of standard questions about the specs and/or operators that have been newly introduced, the questions falling under headings such as "left and right domains", "imps and co-imps", and "natural polymorphism". Two such questions have already been answered for the unit type: it is a monotype and the spec $!$ is an imp. To these we might also add that the function from specs to specs that always returns $\mathbf{1}$ is a relator (because $\mathbf{1}$ is a monotype). This seemingly trivial remark will prove to be quite important. There are two "standard questions" yet to be answered: what are the left and right domains of $!$ and in what sense is it naturally polymorphic? Here is the answer to the first of these.

Theorem 26 $\quad !\langle \ = \ \mathbf{1} \ \text{ and } \ !\rangle \ = \ I$

□

The final question in this list is answered by the following theorem.

Theorem 27 $\quad ! \in \mathbf{1} \leftrightarrow \top\!\!\top$. In particular, for all specs R, $! \in \mathbf{1} \dot{\sim} R$.

□

The second naturality property of $!$ above is much the weaker of the two but may have a more familiar appearance. It is derived from the type statement

$$! \circ A \in \mathbf{1} \longleftarrow A$$

by omitting the restriction of the domain to monotype A (in effect considering the polymorphic imp rather than an instance of it), replacing A by an arbitrary

spec R and replacing "$\longleftarrow$" by "$\rightsquigarrow$". It is this that is often meant by saying that ! is "naturally polymorphic".

The unit type constitutes a building block for the construction of data types; we turn now to the mortar: cartesian product and disjoint sum.

6 Axioms for Cartesian Product and Disjoint Sum

In all systems that we know of, cartesian product and disjoint sum are duals of each other. (Disjoint sum is indeed often given the name "co-product".) In choosing an axiomatisation of the two concepts in a relational framework we have therefore striven for two sets of rules that are "dual" to each other in some clearly recognisable way. It is for this reason that we present the two sets of axioms together in this section. In subsequent sections we consider separately the consequences of the axioms for cartesian product and for disjoint sum before returning in the final section to consider natural isomorphisms between combinations of the two.

In choosing our axioms, we have, of course, been strongly influenced by our experience with set-theoretic presentations of the relational calculus, that being the model our axioms are intended to capture.

We begin by postulating the existence of four specs, for cartesian product the two projections $\ll$ (pronounced "project left") and $\gg$ (pronounced "project right") and for disjoint sum the two injections $\hookrightarrow$ (pronounced "inject left") and $\hookleftarrow$ (pronounced "inject right"). (Note the unconventional direction of the arrow heads. As an aid to memory, and motivation for this choice, we suggest that the reader bear in mind the diagram "$X \hookrightarrow X+Y \hookleftarrow Y$".) Further, experience leads us to introduce four binary operators on specs, for cartesian product $\vartriangle$ (pronounced "split") and $\times$ (pronounced "times"), and for disjoint sum $\triangledown$ (pronounced "junc") and $+$ (pronounced "plus"), defined in terms of the projection and injection specs as follows:

$$P \vartriangle Q = (\ll^{\cup} \circ P) \sqcap (\gg^{\cup} \circ Q) \tag{28}$$

$$P \triangledown Q = (P \circ \hookrightarrow^{\cup}) \sqcup (Q \circ \hookleftarrow^{\cup}) \tag{29}$$

$$P \times Q = (P \circ \ll) \vartriangle (Q \circ \gg) \tag{30}$$

$$P+Q = (\hookrightarrow \circ P) \triangledown (\hookleftarrow \circ Q) \tag{31}$$

The relational model that we envisage assumes that the universe is a term algebra formed by closing some base set under three operators: the binary operator mapping the pair of terms x, y to the term (x, y), and two unary operators $\hookrightarrow$ and $\hookleftarrow$ mapping the term x to the terms $\hookrightarrow.x$ and $\hookleftarrow.x$, respectively. The interpretation of $\ll$ and $\gg$ is that they project a pair onto its left and right components.

Our first axiom is that the injections are both imps.

$$I \sqsupseteq (\hookrightarrow \circ \hookrightarrow^{\cup}) \sqcup (\hookleftarrow \circ \hookleftarrow^{\cup}) \tag{32}$$

The "dual" of this axiom that we propose is:

$$I \sqsupseteq (\ll^{\cup} \circ \ll) \sqcap (\gg^{\cup} \circ \gg) \tag{33}$$

which says that projecting a pair onto its first and second components and then recombining the components leaves the pair unchanged.

(Berghammer and Zierer [4] and de Roever [9] introduce an almost identical axiom to (33) but in their case the axiom is an equality rather than an inclusion. The difference is that their theories are monomorphic and not polymorphic. Relations are assumed to be (externally) typèd and there is a family of product operators indexed by pairs of types. In our theory types (or rather domains) are internal and there is just one (polymorphic) product operator.)

We remark that axioms (32) and (33) take the following form when rephrased in terms of the product and sum operations.

$$I \sqsupseteq I+I \quad \text{and} \quad I \sqsupseteq I \times I \tag{34}$$

This is reassuring since it is one step on the way to guaranteeing that $+$ and $\times$ are binary relators.

Cartesian product and conjunction are closely related. Specifically, we have (in the set-theoretic interpretation of $\times$)

$$x \langle P \cap Q \rangle y \equiv (x,x) \langle P \times Q \rangle (y,y)$$

Abstracting from this property in order to find an axiom that has a pleasing syntactic shape we are led to the following axiom:

$$(P \vartriangle Q)^{\cup} \circ (R \vartriangle S) = (P^{\cup} \circ R) \sqcap (Q^{\cup} \circ S) \tag{35}$$

The dual axiom for disjoint sum is:

$$(P \triangledown Q) \circ (R \triangledown S)^{\cup} = (P \circ R^{\cup}) \sqcup (Q \circ S^{\cup}) \tag{36}$$

As a final axiom we postulate that left projection is possible if and only if right projection is possible:

$$\ll\!\rangle = \gg\!\rangle \tag{37}$$

Property (37) is equivalent $\top\!\!\top \circ \ll = \top\!\!\top \circ \gg$. Its dual is therefore the trivially true $\hookrightarrow \circ \perp\!\!\!\perp = \hookleftarrow \circ \perp\!\!\!\perp$. There are thus no further axioms for disjoint sum.

The five properties (32), (33), (35), (36) and (37) are the sum total of our axiomatisation of cartesian product and disjoint sum. In the following two sections we consider individually the consequences of the axioms for cartesian product and disjoint sum.

7 Properties of Cartesian Product

There is a major complicating factor in developing a *relational* rather than a *functional* theory of datatypes. It is not, however, a complication that we want to avoid or brush under the carpet since it is an inevitable consequence of the desire to face the issue of nondeterminism. The complication can be pinpointed to cartesian product. Consider, as a first example, the "doubling function", i.e. a function that constructs a pair from a singleton by simply copying its argument. This is the imp $I \vartriangle I$. Now consider the equation:

$$(I \vartriangle I) \circ R = R \vartriangle R$$

and let us here interpret R as a *nondeterministic function*. The equation is then clearly invalid since on the left side some nondeterministically calculated value is copied whereas on the right side a pair is constructed by applying R twice; since that calculation is nondeterministic the two elements of the pair may differ. If, however, R is a true function (an imp according to our definition) the equation is valid, as can easily be proved. Clearly the difference lies in the fact that imps distribute backwards over the cap operator whereas that is not the case in general.

The ramifications of the lack of such a distributivity property are manyfold. They can best be observed by comparing the theorems in this section with those in the next. In particular the fusion properties in the subsection 7.1, the computation rules in subsection 7.2 and the terminality property in subsection 7.5 are significantly less tractable than their counterparts for disjoint sum.

7.1 Fusion Properties

Our first concern is whether or not the product operator ($\times$) is a relator. According to the definition of a relator there are four conditions that we must verify. The first condition is axiomatically true (see (34)). The second condition, the requirement that cartesian product be monotonic in both its arguments, is clear from its definition (it is a composition of monotonic functions). Also clear from the definition of cartesian product is that the reverse operator distributes over it. I.e.

$$(P \times Q)^{\cup} \quad = \quad P^{\cup} \times Q^{\cup} \tag{38}$$

It remains to show that composition distributes over cartesian product:

Theorem 39 (Product-Split and Product-Product Fusion)

(a) $(P \times Q) \circ (R \vartriangle S) \quad = \quad (P \circ R) \vartriangle (Q \circ S)$
(b) $(P \times Q) \circ (R \times S) \quad = \quad (P \circ R) \times (Q \circ S)$
□

Properties (39a) and (39b) are the first examples of many properties to which we give the name "fusion" property. In general, whenever we introduce a relator we seek its associated "morphism" operator (in the case of cartesian product this is split, and in the case of disjoint sum this is junc) and we investigate conditions under which two specs can be "fused" into the one morphism. (Typically, as in (39a) and (39b) one of the specs to be fused is itself a morphism.) Elsewhere [3] we discuss relators defined via fixed-points and observe a connection between morphism fusion and loop fusion. Note, however, that we do not always use the rules to "fuse" specs; just as often we use them to "defuse" a spec into component parts. The reader should not allow the one-way character of the name to prejudice their use of such rules.

Corollary 40 $\times$ is a binary relator.
□

A fusion equality in which the split occurs to the left of the composition cannot be established in general. An inclusion does hold, however, and is not

entirely useless. Two cases where an equality can be established (although not the only ones) are when one operand of the split has the form $S \circ \top\top$ for some S and when the right operand of the composition is an imp.

Theorem 41 (Split-Imp Fusion)

(a) $(R \vartriangle S) \circ T \sqsupseteq (R \circ T) \vartriangle (S \circ T)$
(b) $(R \vartriangle (S \circ \top\top)) \circ T = (R \circ T) \vartriangle (S \circ \top\top)$
and, for all imps f,
(c) $(R \vartriangle S) \circ f = (R \circ f) \vartriangle (S \circ f)$
□

7.2 Computation Rules

The name "projection" immediately suggests its operational interpretation. Here that interpretation is represented by two rules that we call "computation rules".

Note that the rules are valid for all specs P and Q but the righthand side of each rule is slightly more complex than a naive examination might suggest.

Theorem 42 (Computation Rules for Split)

(a) $\ll \circ (P \vartriangle Q) = P \circ Q\!>$ (b) $\gg \circ (P \vartriangle Q) = Q \circ P\!>$
□

Theorem 43 (Product is Strict) $R \times S = \bot\!\!\bot \equiv R = \bot\!\!\bot \vee S = \bot\!\!\bot$
□

Theorem 44 (Computation Rules for Product)

(a) $\ll \circ (P \times Q) = P \circ \ll \circ (I \times Q\!>)$
(b) $\gg \circ (P \times Q) = Q \circ \gg \circ (P\!> \times I)$
□

7.3 Imp and Co-imp Preservation

Thus far, our language has implied that the projections are imps but, as yet, we have not stated the fact so explicitly. In fact, more can be said: they are surjective imps.

Lemma 45 $\ll \circ \ll^{\cup} = I$ and $\gg \circ \gg^{\cup} = I$
□

Corollary 46 $\ll$ and $\gg$ are both imps.
□

Theorem 47

(a) $P \vartriangle Q$ is a co-imp $\Leftarrow$ P is a co-imp $\vee$ Q is a co-imp.
(b) $P \vartriangle Q$ is an imp $\Leftarrow$ P is an imp $\wedge$ Q is an imp.
□

Since product is a binary relator we have:

Theorem 48 $\times$ preserves both imps and co-imps.
□

(The fact that a split is a co-imp if just one of its arguments is a co-imp does not help one to prove anything stronger about product.)

7.4 Left and Right Domains

Much of the work necessary to determine the effect of the left and right domain operators on splits and left and right projections has already been completed. Lemma 45, for instance, tells us immediately that the projections are surjective. I.e.

Theorem 49

(a) $\ll\!< = I$ and $\gg\!< = I$
(b) $\ll\!> = I \times I$ and $\gg\!> = I \times I$
□

Since product is a (binary) relator we can immediately instantiate theorem 12 obtaining:

Theorem 50

(a) $(P \times Q)< = P< \times Q<$ (b) $(P \times Q)> = P> \times Q>$
□

We conclude this section with expressions for the right and left domains of a split. That for the left domain is not particularly helpful but is more compact than the expanded form of the definition! (As one might expect it is usually more difficult to predict the left domain than the right domain of a split.)

Theorem 51 (Split Right and Left Domain)

(a) $(P \vartriangle Q)> = P> \sqcap Q>$

(b) $(P \vartriangle Q)< = \ll\cup \circ P \circ Q\cup \circ \gg \sqcap I$
□

7.5 Unique Extension Properties

In the category **Set** of sets and total functions cartesian product is defined via limits of functors, resulting in a so-called "terminality" property. In our system this terminality is valid not only for imps (our equivalent of functions) but also for a more general class of specs (although not for all specs). We refer to the relevant theorem as the "unique extension property" for cartesian product, and it is the purpose of this section to present the property and then to explore some of its consequences. First, a useful lemma.

Lemma 52 $P \vartriangle Q \;=\; (P \circ Q\rangle) \;\vartriangle\; (Q \circ P\rangle)$
□

In its most general form the unique extension property is as follows:

Theorem 53 (Unique Extension Property)
Suppose $\trianglelefteq \in \{\sqsubseteq, =, \sqsupseteq\}$. Assume also that

$$\ll^{\cup} \circ \ll \circ X \;\sqcap\; \gg^{\cup} \circ \gg \circ X \;\trianglelefteq\; X.$$

Then

$$X \trianglelefteq P \vartriangle Q \;\equiv\; \ll \circ X \trianglelefteq P \circ Q\rangle \;\wedge\; \gg \circ X \trianglelefteq Q \circ P\rangle$$
□

More often than not we apply the theorem with the variable "$\trianglelefteq$" instantiated to "$=$". However, since our purpose is to develop a theory that admits program refinement as a possible step we are continually on the lookout for more general properties of the same nature as theorem 53, the cost in terms of burden of proof being typically almost negligible.

The assumption in theorem 53 is somewhat unwieldy; however, it is important to note that it is *not* equivalent to X being an imp. The assumption is indeed quite weak and we shall encounter several instances where it is valid. One such case is where X is itself a split term, resulting in the following cancellation property.

Theorem 54 (Split Cancellation) For all $\trianglelefteq \in \{\sqsubseteq, =, \sqsupseteq\}$

$$P \vartriangle Q \trianglelefteq R \vartriangle S \;\equiv\; P \circ Q\rangle \trianglelefteq R \circ S\rangle \;\wedge\; Q \circ P\rangle \trianglelefteq S \circ R\rangle$$
□

In order to compare the unique extension property with the usual categorical definition of product we restrict X, P and Q to imps. The backwards distribution of imps over intersection shows that the assumption in the statement of the unique extension property is met for imps with left domain in $I \times I$. For the terminality we also have to get rid of the right domains. This explains the assumptions in the terminality theorem.

Theorem 55 (Terminality)
Let f be an imp with $I \times I \sqsupseteq f\langle$ and let $P\rangle = Q\rangle$. Then

$$f \;=\; P \vartriangle Q \;\equiv\; \ll \circ f \;=\; P \;\wedge\; \gg \circ f \;=\; Q$$
□

7.6 Naturality Properties

Part (a) of lemma 39 is a very important property, just as important as part (b). It can be expressed somewhat differently, namely as a naturality property of split.

Theorem 56 (Naturality of Split)

$$\vartriangle \in (R \times S \dot{\leftsquigarrow} T) \dot{\leftsquigarrow} (R \dot{\leftsquigarrow} T) \times (S \dot{\leftsquigarrow} T)$$

□

Note: it is not the case that any of the "$\dot{\leftsquigarrow}$" operators can be replaced by either "$\dot{\leftarrow}$"or "$\dot{\leftrightarrow}$".

Since product is a (binary) relator we can simply instantiate theorem 17 to obtain:

Theorem 57 (Naturality of Product) For all specs R, S, T, U and all $\sim \in \{\dot{\leftsquigarrow}, \dot{\leftarrow}, \dot{\leftrightarrow}\}$

$$\times \in (R \times T \sim S \times U) \dot{\leftsquigarrow} (R \sim S) \times (T \sim U)$$

□

The two projections are also naturally polymorphic in the following sense.

Theorem 58 (Naturality of Left and Right Projection)

For all specs R,
(a) $\ll \in R \dot{\leftrightarrow} R \times \top\!\!\top$ and $\gg \in R \dot{\leftrightarrow} \top\!\!\top \times R$
In particular, for all specs R and S,
(b) $\ll \in R \dot{\leftsquigarrow} R \times S$ and $\gg \in R \dot{\leftsquigarrow} S \times R$
□

(Note that "$\dot{\leftrightarrow}$" in the statement of the theorem can be replaced by "$\dot{\leftsquigarrow}$" or "$\dot{\leftarrow}$" since equality implies inclusion.)

8 Properties of Disjoint Sum

We have discussed the properties of cartesian product before those of disjoint sum because the latter are substantially simpler to derive. This is because the cap operator in the definition of split is replaced by the cup operator in the definition of junc, and composition is universally cup-junctive but not universally cap-junctive. Calculations with split and/or the projections can thus often be transliterated into calculations with junc and/or the injections — but less often the other way round. The order of presentation remains the same so that the reader may compare the properties one-by-one.

8.1 Fusion Properties

As was the case for cartesian product it is straightforward to see that + satisfies three of the conditions necessary for it to be a relator: the first is satisfied axiomatically, and monotonicity and commutation with reverse are satisfied by construction. Distributivity with respect to composition is also a special case of a "fusion" law, namely that a sum can be fused with a junc.

Theorem 59 (Junc-Sum and Sum-Sum Fusion)

(a) $(P \triangledown Q) \circ (R+S) = (P \circ R) \triangledown (Q \circ S)$
(b) $(P+Q) \circ (R+S) = (P \circ R) + (Q \circ S)$
□

Corollary 60 $+$ is a relator.
□

One more fusion property can be added to this list on account of the universal cup-junctivity of composition, namely:

Theorem 61 (Spec-Junc Fusion) $P \circ (Q \triangledown R) = (P \circ Q) \triangledown (P \circ R)$
□

Comparison should be made with theorem 41 where a restriction to imps had to be made in order to obtain an equality.

8.2 Computation Rules

For want of inventiveness we give the name "co-strictness" to the dual of the strictness of product.

Theorem 62 (Co-strictness of Sum) $R+S = \perp\!\!\!\perp \equiv R = \perp\!\!\!\perp \wedge S = \perp\!\!\!\perp$

□

The computation rules for junc do not involve any extra complications (unlike those for split). Their derivation, however, follows the same pattern.

Theorem 63 (Computation Rules for Junc and Sum)

(a) $(P \triangledown Q) \circ \hookrightarrow = P$
(b) $(P \triangledown Q) \circ \hookleftarrow = Q$
(c) $(P+Q) \circ \hookrightarrow = \hookrightarrow \circ P$
(d) $(P+Q) \circ \hookleftarrow = \hookleftarrow \circ Q$
□

8.3 Imp and Co-imp Preservation

Our first axiom was that left and right injection are both imps. In fact they are also co-imps as is evidenced by the following:

Lemma 64 $\hookrightarrow\cup \circ \hookrightarrow = I = \hookleftarrow\cup \circ \hookleftarrow$
□

Corollary 65 $\hookrightarrow$ and $\hookleftarrow$ are bijections.
□

Since split preserves both imps and co-imps one would expect that junc does so too. But this is not the case! The proof that split preserves co-imps cannot be transliterated into a proof that junc preserves co-imps (thus emphasising that one has to be very careful with "dualisation" of arguments) and we can only assert that it preserves imps. Nevertheless, $+$ preserves both.

Theorem 66 (Imp and Co-imp Preservation)

(a) $\triangledown$ preserves imps.
(b) If f and g are both co-imps and $f< \sqcap g< = \perp\!\!\!\perp$ then $f \triangledown g$ is a co-imp.
(c) $+$ preserves both imps and co-imps.
□

8.4 Left and Right Domains

Lemma 64 not only predicts that the injections are co-imps but also that they are total. Formulae for the left domain of the injections are also easy to calculate:

Theorem 67

(a) $\hookrightarrow> = I$ and $\hookleftarrow> = I$
(b) $\hookrightarrow< = I+\perp\!\!\!\perp$ and $\hookleftarrow< = \perp\!\!\!\perp+I$
□

The next theorem could be said to be the dual to the theorem that the right domains of the projections are equal.

Theorem 68 $\hookrightarrow< \sqcap \hookleftarrow< = \perp\!\!\!\perp$
□

In contrast to those for cartesian product the rules for the left and right domains of junc and sum are very simple. Both domain operators distribute over sum, and over junc, but transforming the operator in one case into cup and in the other into sum.

Theorem 69

(a) $(P+Q)> = P> + Q>$ (b) $(P+Q)< = P< + Q<$
(c) $(P \triangledown Q)< = P< \sqcup Q<$ (d) $(P \triangledown Q)> = P> + Q>$
□

8.5 Unique Extension Property

The counterpart of the *terminality* property of cartesian product is an *initiality* property. Here it is yet stronger: so much so indeed that it warrants a different order of presentation. The key insight is that two components in a junc or sum remain truely disjoint. To be precise:

Theorem 70 (Cancellation Properties) For all $\trianglelefteq \in \{\sqsubseteq, =, \sqsupseteq\}$,

(a) $P \triangledown Q \trianglelefteq R \triangledown S \equiv P \trianglelefteq R \wedge Q \trianglelefteq S$
(b) $P+Q \trianglelefteq R+S \equiv P \trianglelefteq R \wedge Q \trianglelefteq S$
□

Corollary 71 (Junc Initiality) For all $\trianglelefteq \in \{\sqsubseteq, =, \sqsupseteq\}$,

$$P \circ (I+I) \trianglelefteq Q \triangledown R \equiv P \circ \hookrightarrow \trianglelefteq Q \wedge P \circ \hookleftarrow \trianglelefteq R$$

□

8.6 Naturality Properties

The naturality properties of the two injections are stronger than those of the projections.

Theorem 72 (Naturality of Left and Right Injection) For all specs R and S,

(a) $\hookrightarrow \in R{+}S \mathrel{\dot{\leftarrow}} R$ (b) $\hookleftarrow \in R{+}S \mathrel{\dot{\leftarrow}} S$

□

The naturality property of junc is also stronger.

Theorem 73 (Naturality of Junc and Sum)
For all specs R, S, T and U and all $\dot{\sim} \in \{\dot{\leftarrow}, \dot{\rightarrow}, \dot{\leftrightarrow}\}$,

(a) $\triangledown \in (R \dot{\sim} S{+}T) \dot{\leftarrow} (R \dot{\sim} S) \times (R \dot{\sim} T)$
(b) $+ \in (R{+}S \dot{\sim} T{+}U) \dot{\leftarrow} (R \dot{\sim} T) \times (S \dot{\sim} U)$

□

9 Natural Simulations and Isomorphisms

To summarise, we now have one non-trivial monotype and two binary relators. Unary relators can be derived from these by fixing one of the arguments to a monotype; ternary relators, quaternary relators etc. can be obtained by composing them in appropriate ways; and new monotypes can be obtained by applying existing relators to existing monotypes. For example, $1{+}1$ and $1 \times (1{+}1)$ are monotypes, and the functions $1+$ and $(1 \times 1)+$ are unary relators. Relators and monotypes built in this way are called *polynomial*. This, however, is just the foundation. It is only now that our theory can really begin.

In this section we make a modest start to showing the ease with which certain calculations can be made within the theory. At the same time we formulate a number (at this point in time 2!) of concepts of particular relevance to data reification.

Definition 74 (Natural Simulation) Relator F is said to (naturally) simulate relator G if and only if there exists a spec γ such that

(a) for all specs R, $\gamma \in F.R \mathrel{\dot{\leftarrow}} G.R$
(b) γ is an imp
(c) $\gamma{>} = G.I$

The spec γ itself is called a natural simulation.

□

Note that the combination of conditions (a) and (b) implies that $\gamma{<} \sqsubseteq F.I$.

Definition 75 (Natural Isomorphism) Relators F and G are said to be naturally isomorphic if and only if there exists a spec γ such that

(a) for all specs R, $\gamma \in F.R \mathrel{\dot{\leftarrow}} G.R$
(b) γ is a bijection
(c) $\gamma{<} = F.I$ and $\gamma{>} = G.I$

The spec γ itself is called a natural isomorphism.
□

Examples of natural isomorphisms are provided by the two injections $\hookrightarrow$ and $\hookleftarrow$. The former is a natural isomorphism between the relator $(+\!\!\perp\!\!\!\perp)$ and the identity relator. I.e.

$\hookrightarrow \in R+\!\!\perp\!\!\!\perp \dot{\leftrightarrow} R$, for all specs R
$\hookrightarrow$ is a bijection, and
$\hookrightarrow^{<} = I+\!\!\perp\!\!\!\perp$ and $\hookrightarrow^{>} = I$

Similarly, the latter is an isomorphism between the the relator $(\perp\!\!\!\perp+)$ and the identity relator. The injections are also examples of natural simulations: $\hookrightarrow$ is, for example, a natural simulation of the identity relator by the relator $+\mathbf{1}$. (In general any monotype may be used in place of $\mathbf{1}$.)

As might be expected, both natural simulations and natural isomorphisms enjoy many simple but powerful algebraic properties. In later versions of this paper it is our intention to document some of them. For the time being, however, we leave the reader the pleasure of their discovery. Let us proceed to more significant examples. We begin with the most complicated, basic example of a natural isomorphism.

Consider the ternary relators defined by

$$\begin{array}{lcl} R,S,T & \mapsto & R \times (S+T) \\ R,S,T & \mapsto & (R \times S)+(R \times T) \end{array}$$

Our objective is to show that the two relators are naturally isomorphic.

To complete this task we must exhibit a spec, γ, satisfying three quite strong conditions. We can make progress in this task by temporarily setting aside two of the conditions, *constructing* γ to satisfy the remaining condition, and then (hopefully) *verifying* that it satisfies the two other conditions. The condition singled out should be the one that leaves the least freedom to manoeuvre, in this case clearly condition (a). What we shall now demonstrate is how systematically this can be done using the rules we have given for the naturally polymorphic type of the operators we have introduced.

Here then is the construction of the desired natural isomorphism. Assume R, S, and T are arbitrary specs. Then

by construction of γ:

$$\begin{array}{ll} & \gamma \in R \times (S+T) \dot{\leftrightarrow} (R \times S)+(R \times T) \\ \Leftarrow & \quad \{ \text{ naturality of } \triangledown,\ \gamma := \gamma_1 \triangledown \gamma_2 \ \} \\ & \quad \gamma_1 \in R \times (S+T) \dot{\leftrightarrow} R \times S \\ & \wedge\ \gamma_2 \in R \times (S+T) \dot{\leftrightarrow} R \times T \\ \Leftarrow & \quad \{ \text{ naturality of product, } I \in R \dot{\leftrightarrow} R, \\ & \qquad \gamma_1 := I \times \gamma_1,\ \ \gamma_2 := I \times \gamma_2 \ \} \\ & \quad \gamma_1 \in S+T \dot{\leftrightarrow} S \ \wedge\ \gamma_2 \in S+T \dot{\leftrightarrow} T \\ \Leftarrow & \quad \{ \text{ naturality of the injections } \} \\ & \quad \gamma_1 = \hookrightarrow \ \wedge\ \gamma_2 = \hookleftarrow \end{array}$$

Thus the constructed spec is γ where

$$\gamma \;=\; (I \times \hookrightarrow) \triangledown (I \times \hookleftarrow)$$

It remains to show that γ is a bijection and has the correct left and right domains. The verifications are straightforward, but we give them nonetheless as proof of the pudding.

First, we assert that γ is a bijection. That it is an imp follows because it is built out of imps using imp-preserving operators. Since junc is not necessarily co-imp preserving we need to take further steps to show that it is a co-imp.

$$\begin{array}{ll}
 & \gamma \text{ is a co-imp} \\
\Leftarrow & \quad \{ \text{ theorem 66(b) } \} \\
 & (I \times \hookrightarrow \text{ and } I \times \hookleftarrow \text{ are co-imps }) \\
 & \wedge\ (I \times \hookrightarrow)^{<} \sqcap (I \times \hookleftarrow)^{<} = \perp\!\!\!\perp \\
\equiv & \quad \{ \text{ theorem 47 } \} \\
 & (I \times \hookrightarrow)^{<} \sqcap (I \times \hookleftarrow)^{<} = \perp\!\!\!\perp \\
\equiv & \quad \{ \text{ theorem 50 } \} \\
 & (I \times \hookrightarrow^{<}) \sqcap (I \times \hookleftarrow^{<}) \;=\; \perp\!\!\!\perp \\
\equiv & \quad \{ \text{ distributivity property in full paper } \} \\
 & I \times (\hookrightarrow^{<} \sqcap \hookleftarrow^{<}) \;=\; \perp\!\!\!\perp \\
\equiv & \quad \{ \text{ theorem 68, product is strict } \} \\
 & \textbf{true}
\end{array}$$

We now calculate the left domain of γ.

$$\begin{array}{ll}
 & \gamma^{<} \\
= & \quad \{ \text{ definition of } \gamma, \text{ theorem 69(c) } \} \\
 & (I \times \hookrightarrow)^{<} \;\sqcup\; (I \times \hookleftarrow)^{<} \\
= & \quad \{ \text{ theorems 50 and 67 } \} \\
 & I \times (I + \perp\!\!\!\perp) \;\sqcup\; I \times (\perp\!\!\!\perp + I) \\
= & \quad \{ \text{ distributivity property in full paper } \} \\
 & I \times ((I + \perp\!\!\!\perp) \sqcup (\perp\!\!\!\perp + I)) \\
= & \quad \{ \text{ definition of } + \} \\
 & I \times (I + I)
\end{array}$$

Finally, we calculate the right domain of γ.

$$\begin{array}{ll}
 & \gamma^{>} \\
= & \quad \{ \text{ definition of } \gamma, \text{ theorem 69(d) } \} \\
 & (I \times \hookrightarrow)^{>} \;+\; (I \times \hookleftarrow)^{>} \\
= & \quad \{ \text{ theorems 50 and 67 } \} \\
 & (I \times I) + (I \times I)
\end{array}$$

This completes the verification.

The point of discussing this example in so much detail is to emphasise the importance of type considerations in *constructing* specs having prescribed properties. (This is a somewhat different emphasis than that which one encounters most frequently. Wadler [11], for example, discusses the use of natural polymorphism to *infer* properties of already constructed functions.) There is, however, yet more that can be said about the bijection γ that we have constructed that

so far as we know is not predicted by any naturality theorem and yet seems "obvious" from type considerations. The properties that we allude to record its behaviour with respect to the two morphisms split and junc. Before stating the properties we need to interpose a truly remarkable and elegant law permitting an exchange of split for junc and vice-versa.

Theorem 76 (Split-Junc Abide Law)

$$(R \triangledown S) \vartriangle (T \triangledown U) \quad = \quad (R \vartriangle T) \triangledown (S \vartriangle U)$$

□

The properties of the natural isomorphism γ that we anticipated above can now be given.

Theorem 77

(a) $\gamma \circ (R \vartriangle S)+(R \vartriangle T) \quad = \quad (R \circ I \triangledown I) \vartriangle (S+T)$
(b) $R \times (S \triangledown T) \circ \gamma \quad = \quad (R \times S) \triangledown (R \times T)$
□

Natural isomorphisms seem to receive scant attention in the category theory literature, often being relegated to a brief exercise. This is somewhat unfortunate because it deemphasises their importance and it means that no guidance is given on how to construct them. Unfortunately space limitations have denied us the opportunity to offer such guidance and we make do with the following summary of the elementary natural isomorphisms. It is possible, however, with the aid of the information documented here, to construct the all isomorphisms. Otherwise the reader may consult the full paper. In order to list the isomorphic relators we use a home-grown, but hopefully self-evident, lambda notation.

$$\lambda(R :: \ \bot\!\!\!\bot) \ \cong \ \lambda(R :: \ R \times \bot\!\!\!\bot) \tag{78}$$
$$\lambda(R :: \ R+\bot\!\!\!\bot) \ \cong \ \lambda(R :: \ R) \tag{79}$$
$$\lambda(R, S :: \ R+S) \ \cong \ \lambda(R, S :: \ S+R) \tag{80}$$
$$\lambda(R, S, T :: \ R+(S+T)) \ \cong \ \lambda(R, S, T :: \ (R+S)+T) \tag{81}$$
$$\lambda(R, S :: \ R \times S) \ \cong \ \lambda(R, S :: \ S \times R) \tag{82}$$
$$\lambda(R, S, T :: \ R \times (S \times T)) \ \cong \ \lambda(R, S, T :: \ (R \times S) \times T) \tag{83}$$
$$\lambda(R :: \ R) \ \cong \ \lambda(R :: \ R \times \mathbf{1}) \tag{84}$$
$$\lambda(R, S, T :: \ R \times (S+T)) \ \cong \ \lambda(R, S, T :: \ (R \times S)+(R \times T)) \tag{85}$$

The following properties of the constructed isomorphisms are also of interest. (The names $\alpha_1 \ldots \alpha_8$ have been given to the isomorphisms in order of their appearance in the list above.)

$$\bot\!\!\!\bot \quad = \quad \alpha_1 \circ R \vartriangle \bot\!\!\!\bot \tag{86}$$
$$R \triangledown \bot\!\!\!\bot \circ \alpha_2 \quad = \quad R \tag{87}$$
$$R \triangledown S \circ \alpha_3 \quad = \quad S \triangledown R \tag{88}$$
$$R \triangledown (S \triangledown T) \circ \alpha_4 \quad = \quad (R \triangledown S) \triangledown T \tag{89}$$
$$R \vartriangle S \quad = \quad \alpha_5 \circ S \vartriangle R \tag{90}$$
$$R \vartriangle (S \vartriangle T) \quad = \quad \alpha_6 \circ (R \vartriangle S) \vartriangle T \tag{91}$$
$$R \quad = \quad \alpha_7 \circ R \vartriangle ! \tag{92}$$

Note the pattern: the relators $+$ and $\times$ have been systematically replaced by their corresponding morphism and $\mathbf{1}$ has been replaced by its morphism $!$. The α's are eaten up on the right side by a junc and on the left side by a split.

This concludes our discussion of natural simulations and of the elementary properties of the polynomial relators.

References

[1] R.C. Backhouse. On a relation on functions. In W.H.J. Feijen, A.J.M. van Gasteren, D. Gries, and J. Misra, editors, *Beauty is our Business*. Springer-Verlag, 1990.

[2] R.C. Backhouse, Bruin P. de, P. Hoogendijk, G. Malcolm, Voermans T.S., and J. van der Woude. A relational theory of datatypes. In preparation: copies of draft available on request, 1991.

[3] R.C. Backhouse, Bruin P. de, G. Malcolm, Voermans T.S., and J. van der Woude. Relational catamorphisms. In Möller B., editor, *Proceedings of the IFIP TC2/WG2.1 Working Conference on Constructing Programs*. Elsevier Science Publishers B.V., 1991.

[4] R. Berghammer and H. Zierer. Relational algebraic semantics of deterministic and nondeterministic programs. *Theoretical Computer Science*, 43:123–147, 1986.

[5] P.J. de Bruin. Naturalness of polymorphism. Technical Report CS8916, Department of Mathematics and Computing Science, University of Groningen, 1989.

[6] E.G. Manes and M.A. Arbib. *Algebraic Approaches to Program Semantics*. Texts and Monographs in Computer Science. Springer-Verlag, Berlin, 1986.

[7] L. Meertens. Constructing a calculus of programs. In J.L.A. van de Snepscheut, editor, *Conference on the Mathematics of Program Construction*, pages 66–90. Springer-Verlag LNCS 375, 1989.

[8] J.C. Reynolds. Types, abstraction and parametric polymorphism. In R.E. Mason, editor, *IFIP '83*, pages 513–523. Elsevier Science Publishers, 1983.

[9] Willem Paul de Roever Jr. *Recursive program schemes: semantics and proof theory*. PhD thesis, Free University, Amsterdam, January 1974.

[10] G. Schmidt and T. Ströhlein. Relation algebras: Concept of points and representability. *Discrete Mathematics*, 54:83–92, 1985.

[11] P. Wadler. Theorems for free! In *4'th Symposium on Functional Programming Languages and Computer Architecture, ACM, London*, September 1989.