GEOMETRIC LOGIC AS A SPECIFICATION LANGUAGE

STEVEN VICKERS
Department of Computing, Imperial College
180 Queen's Gate, London SW7 2BZ, United Kingdom
E-mail: sjv@doc.ic.ac.uk

ABSTRACT

The "observational content" of geometric logic is discussed and it is proposed that geometric logic is an appropriate basis for a Z-like specification language in which schemas are used as geometric theory presentations.

A descriptional mechanism of "schema entailment", generalizing type constructions and logical entailment, is defined and investigated in some examples, and is also used in defining schema morphisms which are discussed briefly in connection with schema connectives, and with specifying and implementing operations.

1 Introduction

In Vickers [10, 9] it was suggested rather briefly that an "observational content" of geometric logic made it a good candidate as a logic appropriate to specification. The aim of this paper is to amplify those remarks.

What is Specification?

Fundamentally, a specification must attempt to capture the relation between the computer system and the real world, for that relation is what gives meaning to the electronic bit manipulations. In the words of Maibaum and Turski [8], the specification binds together a program and its application. Of course, the real world is ultimately informal, so that no formal system can ever truly bind to it. Nonetheless, in writing our specification we should attempt to formalize as best we can the application domain, that is to say those aspects of the real world that are relevant to the meaning of the computer system. Thus logically, though not teleologically, the specification binds together two formal systems: a formal theory of our view of the real world, and the formal theory of the computers.

What Sort of Logic?

Having accepted that our specification is to include an attempt at formalizing the application domain, we should try to use a logic that is sympathetic to the real world and does not try to foist on it artifacts of the formalism. To illustrate what I mean, consider the formula

$$\exists\, x \bullet (\mathit{LivesIn}(x, \mathit{LochNess}) \wedge \mathit{Monster}(x))$$

Assuming that the predicates *LivesIn* and *Monster* are interpreted approximately in accordance with the corresponding English words, we can well imagine under what conditions we could know that the assertion is true (Figure 1). To be sure, this would take a lot of luck, serendipity or Divine Grace, but no particular effort – in fact we can imagine others putting much time and resources into a vain attempt (Figure 2) to repeat our observation. And after our revelation we at least can be in no doubt about the truth of the assertion.

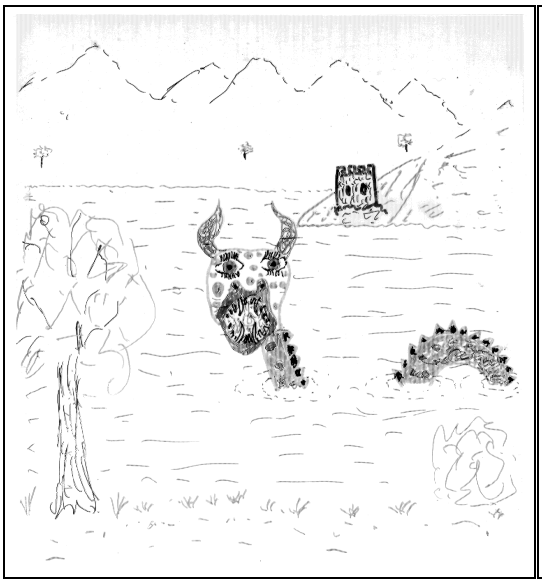


Figure 1

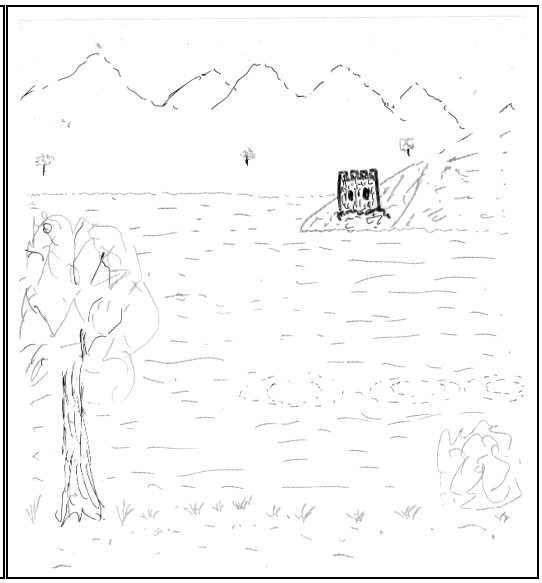


Figure 2

But now consider its formal negation,

$$\forall\, x \bullet (\neg\, \mathit{LivesIn}(x, \mathit{LochNess}) \vee \neg\, \mathit{Monster}(x))$$

There is no conceivable revelation of the truth of *this* assertion, and to check it by personal effort is quite impracticable. On the face of it we'd have to look at every thing and check that either it doesn't live in Loch Ness or that it's not a monster; and even more economical approaches such as draining the Loch and sifting through the muck at the bottom would be too difficult in practice.

The moral is that negation is not the inoffensive notion that classical logic would make it appear, for it can completely change the character of an assertion to which it is applied. In fact we shall work with a positive logic, without negation.

One consequence of this is that it is much less meaningful to ask of an assertion, "Is this true or false?" Instead, one should ask, "What truth can I find in this?"

Observational Classes

To give a structure for a many-sorted classical theory, one interpretes each sort as a set (its carrier) and each predicate – indeed, each formula – as a relation, a subset of the appropriate product of carriers (corresponding to the free variables). Hence formulae are represented as sets.

That is a mathematical semantics, but it is intended to indicate a real world (informal) meaning in that set-theoretic operations are considered in themselves to have a clear meaning in the real world: set comprehension is "gathering things together" (though one might question how meaningful this is for infinite sets).

We have no ready-made mathematical model for our observational notions, but let us at least briefly recall the real world intuitions as set out in [10]. A formula is to be interpreted as an "observational class" which comprises two rules:

- How to know when you have "apprehended" an element of the class – *i.e.* observed its existence and got a hold on it for comparison with other elements.
- How to know when you have observed that two apprehended elements are equal.

(The slightly fussy "how to know when you have." is intended to stress the serendipitous nature of observations that has already been remarked on.)

I shouldn't pretend that this account of observational classes arises fully formed out of a fundamental analysis of the nature of observations – on the contrary, it is undeniably influenced by the existence of geometric logic with its known nice mathematics. One can certainly question some of the observational assumptions. For instance, do we really always know how to observe equality? How do you apprehend a photon in such a way that you can observe its equality with another photon? Why shouldn't inequality be fundamental? Nonetheless, I believe the principles described here lead to specifications that are at least better adapted to observational reality than those based on classical logic (or intuitionistic logic, for that matter).

Schemas as Theory Presentations

In a theory presentation (which will be a schema in the Z-like language we shall propose) the application domain, our view of the world, is to be described as follows.

- An extralogical vocabulary of sorts (base types), predicates and functions represents the observational classes that we believe (informally) exist in the application domain. Derived formulae will represent derived observational classes. This will be described more fully in the next section, but remember that the connectives are not always those of classical logic – no negation, for instance.

- A set of axioms represents our background assumptions or hypotheses about the real world. An axiom is not itself a formula (observational class), but compares two formulae by asserting that if one observation is made, then another is also possible.

As an example, the application domain for a genealogy database clearly includes notions of people and parenthood. We might start to present their theory as a schema

$$
\begin{array}{|l}
\hline
\mathit{People}[X] \\
\hline
ma, pa : X \to X \\
\neq : \mathbb{P}(X \times X) \\
\hline
x \neq x \Longrightarrow_{x:X} \text{false} \\
\text{true} \Longrightarrow_{x,y:X} x = y \lor x \neq y \\
ma(x) = pa(x) \Longrightarrow_{x:X} \text{false} \\
\bigwedge_{i=1}^{n} (x_i = ma(x_{i-1}) \lor x_i = pa(x_{i-1})) \\
\quad \land (x_0 = ma(x_n) \lor x_0 = pa(x_n)) \Longrightarrow_{x_0,\ldots,x_n:X} \text{false} \\
\hline
\end{array}
$$

- "People" is the name of the theory.

- X is the base type whose elements are people; it appears in the position that in Z is used for "generic types". Hence our account explains generic types as sorts in a many-sorted theory.

- The extralogical vocabulary appears in the upper part of the schema, which Z calls the "signature", while the axioms appears in the lower, "predicate" part. However, for us a "predicate" will be a predicate symbol (like $\neq$) in the vocabulary.

- We can observe when two people are *different* people, but since inequality (unlike equality) is not part of the logic it must be supplied as an extralogical ingredient of the theory, together with axioms (the first two) to characterize it as the complement of equality.

  A type that is equipped with inequality as well as equality is said to be *decidable*.

- *ma* and *pa* are to be total functions. Note that they could be equivalently presented as binary relations with axioms for totality and single-valuedness:

$$\begin{array}{|l} ma : \mathbb{P}(X \times X) \\ \hline \text{true} \Longrightarrow_{x,y:X} \exists\, y \bullet ma(x,y) \\ ma(x,y) \wedge ma(x,y') \Longrightarrow_{x,y:X} y = y' \end{array}$$

  and similarly for *pa*.

- People here are intended to be *real people*, not representations in the database – the database will be described on top of the theory of real people (as a schema that includes People).

- The third and fourth axioms are included for illustration – they are not intended to be complete. The third says that nobody can be both a mother and a father, while the fourth – a countable set of axioms, indexed by $n > 1$ – says that no one can be their own ancestor.

Actually, we have already now encountered mathematical difficulties. Clearly the database is finite, and we cannot expect *ma* and *pa* to restrict to total functions on the database – we can't expect to know the parents of everyone. However, mathematically speaking, the axiom of non-self-ancestrality is sufficient to rule out any finite models (except the empty set) and that should include the set of real-world people – surely that is finite? Certainly we can give undoubted upper bounds – a trillion trillion, say – for the number of people to date (and to interpret *ma* and *pa* we don't need to consider future people), though perhaps it's unreasonable to expect a 1-1 function from the set of people to the set of the first trillion trillion natural numbers. One might think that the theory must be modified in consequence – perhaps *ma* and *pa* are not quite total, everyone having a mother and a father except for Adam and Eve. Yet I cannot believe that this makes better observational sense than the theory given.

A sharp question is the following: Can we accept that there is a 1-1 function from the natural numbers $\mathbb{N}$ to the set of people, defined by $n \mapsto ma^n(S.J.Vickers)$? It is difficult to see how experience could conflict with this idea. Allowing for

30 years per generation, reason would suggest that $ma^{75}(S.J.Vickers)$ is a well-defined person from the classical world, yet the chances of ever identifying her are negligible even for this small value of n. Perhaps one can argue from this example that the world is *not* logically classical.

We shall use the term *schema* synonymously with (geometric) theory presentation. Of course, this is tendentious. Z-schemas are not generally understood as theory presentations, and in fact some of the uses for schemas in Z are not really compatible with this idea. We shall not use the standard Z-terminology for components of schemas:

Z	*Here*
generic type	base type
signature	functions, constants and predicates
generic types + signature	vocabulary
predicate	axioms
name with power set type	predicate

2 Geometric Theories

The logic we shall use is geometric logic, and our Z-like specification language will be called, provisionally, GeoZ.

An introduction to the basic notions of geometric logic is given in [9]. Let us now recall the definitions and explore the application to specifications.

The Basic Form of Presentations

The basic definition of a geometric theory is as follows: it comprises –

- base types (or sorts)
- function and predicate symbols
- axioms

The base types and function and predicate symbols should not cause any surprise to anyone familiar with first-order many-sorted logic. Each function or predicate has an *arity* describing the types of its arguments and (for a function) that of its result, so arities take the form $\mathbb{P}(X_1 \times \cdots \times X_n)$ (for predicates) or $X_1 \times \cdots \times X_n \to X$ (for functions) where the X_i's and X are types. If

$$P : \mathbb{P}(X_1 \times \cdots \times X_n) \qquad \text{and} \qquad f : X_1 \times \cdots \times X_n \to X$$

and if each t_i is a well-formed term of type X_i, then $P(t_1, \ldots, t_n)$ is a well-formed formula and $f(t_1, \ldots, t_n)$ is a well-formed term, of type X. Note the nullary cases, $n = 0$: then P is a *propositional* symbol and f is a *constant*.

Examples:

$less : \mathbb{P}(X \times X)$
$snowing : \mathbb{P}(1)$ (a propositional symbol)
$plus : X \times X \to X$
$zero : X$ (a constant – could write zero : $1 \to X$)

In Z, the arities $\mathbb{P}(X_1 \times \cdots \times X_n)$ and $X_1 \times \cdots \times X_n \to X$ are considered to be types. This is not the case in GeoZ, where there is a firm distinction between types and arities. This is enforced mathematically, but the intuition is that types represent observational classes whereas arities need not: there is no general prescription for apprehending functions or possibly infinite subsets, nor for observing (in a finite way) equality between them. Since we haven't described any type constructions yet (though they will come), the types X_i and X above must be base types in the basic form of presentation described here.

When we come to the axioms, the geometric logic begins to make a big difference. A geometric axiom has the form $\phi \Longrightarrow_S \psi$ ($\phi \vdash_S \psi$ in [9]) where

- S is a finite set of typed variables
- ϕ and ψ are geometric formulae whose free variables are all in S.

Geometric formulae are constructed in an unsurprising way except that a particular set of logical connectives is used. First, terms are constructed from variables (typed) and function symbols in the usual way. Then formulae ϕ are of the form
$\phi ::= t_1 =_X t_2 \mid P(t_1, \ldots, t_n) \mid \text{false} \mid \text{true} \mid \phi \wedge \psi \mid \phi \vee \psi \mid \bigvee_{i \in I} \phi_i \mid \exists\, x : X \bullet \phi$

- In the equation $t_1 =_X t_2$, t_1 and t_2 must be of type X (which in general can safely be omitted).
- In $P(t_1, \ldots, t_n)$, the types of the t_i's must conform with the arity of P.
- In $\bigvee_{i \in I} \phi_i$, I can be *any* indexing set, possibly infinite.
- Despite the possibility of infinite disjunctions, a geometric formula must have only finitely many free variables. However, it can have infinitely many bound ones.

To illustrate quickly the observational content of the geometric connectives, consider $\exists\, x : X \bullet \phi(x, y)$ where $y : Y$ (X and Y are types, not arities). ϕ is to represent an observational subclass of $X \times Y$: to apprehend an element of it you must apprehend elements x and y of X and Y, and make some additional observation to correspond to (x, y) being in ϕ. To observe that $(x, y) = (x', y')$,

you must observe that $x = x'$ and $y = y'$. Now $\exists\, x : X \bullet \phi(x, y)$ is to represent an observational subclass of Y. To apprehend an element of it, you must apprehend a pair (x, y) in ϕ, and to observe that $(x, y) = (x', y')$ you must observe that $y = y'$ – so the only difference between $\phi(x, y)$ and $\exists\, x : X \bullet \phi(x, y)$ lies in the equality.

Now consider an axiom $\phi \Longrightarrow_X \psi$ where $X = \{x_1, x_2, \ldots, x_n\}$, and each x_i has type X_i. ϕ and ψ then represent observational subclasses of $\Pi_i X_i$ (this is the case even when not all the x_i's appear explicitly in ϕ or ψ as free variables) and the axiom asserts that the subclass for ϕ is contained in that for ψ: whenever you have apprehended elements x_i and made the extra observations for ψ, then the extra observations for ψ are also possible (even if you haven't made them yet). The axiom is not itself observable but asserts a hypothesis about the way the observational classes relate.

Extensions to the Logic

Plainly the licence to use infinitary disjunctions is a doubtful privilege. You might expect in practice to be limited to finitary disjunctions by rather mundane considerations such as availability of paper (unless a formal set theory is built into the logic, and we're not going to do that). Unfortunately, a restriction to finitary disjunctions is too restrictive for practical purposes. However, there is the possibility of using finitary notation for fragments of the infinitary logic, and we shall use this technique extensively. It's important to understand this correctly. GeoZ is not defined by a particular collection of finitary constructions, but by infinitary constructions for which we shall present some finitary approximations that can be extended by "expert" users. (Non-expert users would use standard packages.)

That's rather vague. We shall see more specifically two regions where extensions can be made: in the type system and in the logical connectives. An important principle that applies to both is that they should *not* increase the expressive power of the infinitary logic: the theories that can be presented using the extensions should be already presentable using the infinitary logic.

Extended Types

We have already said very firmly that $\mathbb{P}$ and $\rightarrow$ are used to construct *arities*, and not types. For solid mathematical reasons, they cannot be considered geometric type constructors. However, there are plenty of useful type constructors that *are* geometric. One is Cartesian product, which we have already seen used in arities. Other ones are disjoint union (coproduct), finite power sets, and list types. We shall give a fuller list later.

The fundamental rule is as follows: a type construction can be considered geometric if it can be characterized uniquely up to isomorphism using geometric axioms.

To see how this works, consider Cartesian products. Let X and Y be types in some theory S, and suppose we extend S to a theory T by adding –

- another base type Z
- functions $\pi_1 : Z \to X$ and $\pi_2 : Z \to Y$
- axioms –

$$\begin{array}{l} \mathrm{true} \Longrightarrow_{x:X;\, y:Y} \exists\, z : Z \bullet (\pi_1(z) = x \wedge \pi_2(z) = y) \\ \pi_1(z) = \pi_1(z') \wedge \pi_2(z) = \pi_2(z') \Longrightarrow_{z,z':Z} z = z' \end{array}$$

Given any model of S, we can make a model of T by interpreting Z as $X \times Y$ and π_1 and π_2 as the projection functions. But actually this is "essentially" (*i.e.* up to isomorphism) forced on us: in any model of T, Z has to be isomorphic to $X \times Y$ and π_1 and π_2 have to be isomorphic to the projections. Hence the models of T are essentially the same as those of S, and the two theories are equivalent. Other constructions that can be characterized geometrically in this way include disjoint unions (coproducts; even infinitary coproducts), colimits, finite limits and free algebras. Let us give a provisional, incomplete syntax of types X and arities α:

$$\begin{array}{lcl} X & ::= & \langle \text{base type} \rangle \mid X \times X \mid X \uplus X \mid \mathbb{F}\, X \mid \mathrm{seq}\, X \mid \phi \mid \cdots \\ \alpha & ::= & X \mid \mathbb{P}(X) \mid X \to X \end{array}$$

$\uplus$ is disjoint union, $\mathbb{F}$ is finite power set (free semilattice) and $\mathrm{seq}\, X$ is a list type (finite lists over X – free monoid). We have simplified arities by using the fact that Cartesian product is a type constructor.

The ϕ is a geometric formula. If its free variables $x_1, \ldots, x_n$ have types $X_1, \ldots, X_n$, then ϕ is a *subtype* of $X_1 \times \cdots \times X_n$. There are various naughtinesses here. We have assumed that $\times$ is associative and omitted brackets in $X_1 \times \cdots \times X_n$, but of course it is actually associative only up to isomorphism. Also, there is no reason why there should be a canonical order on the free variables of ϕ, so we have even in effect assumed that $\times$ is commutative. Finally, it would actually be better to allow ϕ to have "unused" free variables (as happens in an axiom $\phi \Longrightarrow_X \psi$ – some of the variables in X may be unused in ϕ and ψ). So all in all there is some formal imprecision in our use of formulas as types, but in practice we shall try not to leave any serious unresolved ambiguities.

3 Schema Entailment

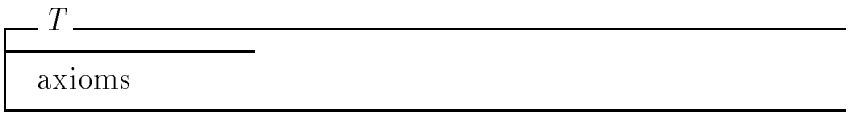
We propose a particular definitional method based on the idea of two theories having "essentially the same models". It subsumes derived types, derived terms and derived axioms (*i.e.* logical consequences).

Let S be a (geometric) schema, and let T be an *extension* for S: by this we mean that it comprises base types, vocabulary and axioms that become a schema when S is included. Or, to put it another way, T is understood in a context that implicitly includes S, so that for instance the arities in T may refer back to S. We write $S + T$ for the schema formed by including S in T (or extending S by T), a superschema of S.

Clearly every model of $S + T$ yields a model of S by reduction. We say that T *definitionally* extends S, and write $S \vdash T$, a *schema entailment*, iff reduction gives an equivalence between models of $S + T$ and models of S – so $S + T$ has essentially the same models as S. (Remember that "essentially" here means "up to isomorphism".) Hence for any model of S there is one and essentially only one way of interpreting the extra ingredients of T in it. (We also say that $S + T$ is an *equivalence superschema* of S.)

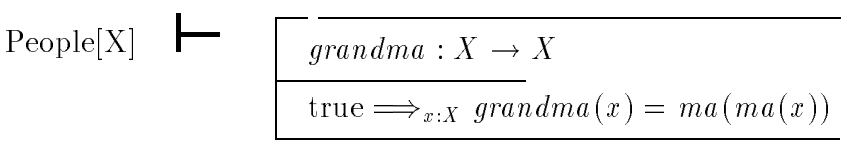
Examples

1. Suppose T contains axioms only:

   Then $S \vdash T$ iff the axioms in T are consequences of the theory S. (Otherwise the axioms of T are proper constraints on S that are not satisfied by every model of S. Hence $S + T$ has fewer models.)

2. Extensions by pure definition are easily seen to be definitional in the sense just given. For instance,

People[X] $\vdash$

$$\begin{array}{|l}\hline \mathit{fullsib} : \mathbb{P}(X \times X) \\ \hline \mathit{fullsib}(x, y) \Longleftrightarrow_{x,y:X} \\ \quad ma(x) = ma(y) \land pa(x) = pa(y) \\ \hline \end{array}$$

3. We can use a similar technique for less direct definitions. For instance, here we define a function from its graph, a total, single-valued relation, thus gaining access to the term-forming mechanisms in the syntax of functions.

$$\begin{array}{|l}\hline [X, Y] \\ P : \mathbb{P}(X \times Y) \\ \hline \text{true} \Longrightarrow_{x:X} \\ \quad \exists\, y : Y \bullet P(x, y) \\ P(x, y) \land P(x, y') \\ \quad \Longrightarrow_{x:X;\ y,y':Y} y = y' \\ \hline \end{array} \vdash \begin{array}{|l}\hline f : X \to Y \\ \hline y = f(x) \Longleftrightarrow_{x:X;\ y:Y} P(x, y) \\ \hline \end{array}$$

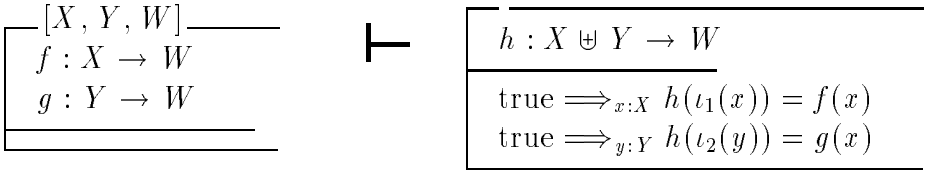
Here is another example to show how a predicate can be considered as a type.

$$\begin{array}{|l}\hline [X] \\ P : \mathbb{P}\, X \\ \hline \\ \hline \end{array} \vdash \begin{array}{|l}\hline [Y] \\ i : Y \to X \\ \hline i(y) = i(y') \Longleftrightarrow_{y,y':Y} y = y' \\ P(x) \Longleftrightarrow_{x:X} \exists\, y : Y \bullet x = i(y) \\ \hline \end{array}$$

4. Definitional extensions are also used to characterize derived types, for instance (a):

$$\begin{array}{|l}\hline [X, Y] \\ \\ \\ \hline \end{array} \vdash \begin{array}{|l}\hline [Z\ [\equiv X \uplus Y]] \\ \iota_1 : X \to Z \\ \iota_2 : Y \to Z \\ \hline \iota_1(x) = \iota_2(y) \Longrightarrow_{x:X;\ y:Y} \text{false} \\ \iota_1(x) = \iota_1(x') \Longrightarrow_{x,x':X} x = x' \\ \iota_2(y) = \iota_2(y') \Longrightarrow_{y,y':Y} y = y' \\ \text{true} \Longrightarrow_{u:Z} \exists\, x : X \bullet u = \iota_1(x) \\ \qquad \lor \exists\, y : Y \bullet u = \iota_2(y) \\ \hline \end{array}$$

This characterizes – it turns out – binary coproducts; in fact, more precisely we have (b):



These schema entailments are actually serving two purposes. Semantically, they assert essentially unique existence, while syntactically they introduce notation. For example, the notation $\uplus$ for disjoint union, introduced in 4(a), is then used in 4(b). By this use, the conclusion of 4(a) is implicitly included in that of 4(b).

By no means all of the examples given are intended to set up notation as well as assert essentially unique existence. To exploit (3), for instance, if we are given a total, single-valued relation P then its corresponding function would not always be called f. Rather, we should say something along the lines of "let $g45 : X \to Y$ be the unique function satisfying $y = g45(x) \Longleftrightarrow P(x, y)$" (or, "let $g45 : X \to Y$ be the function whose graph is P).

Hence further work is needed to clarify the way that definitional extensions are used syntactically.

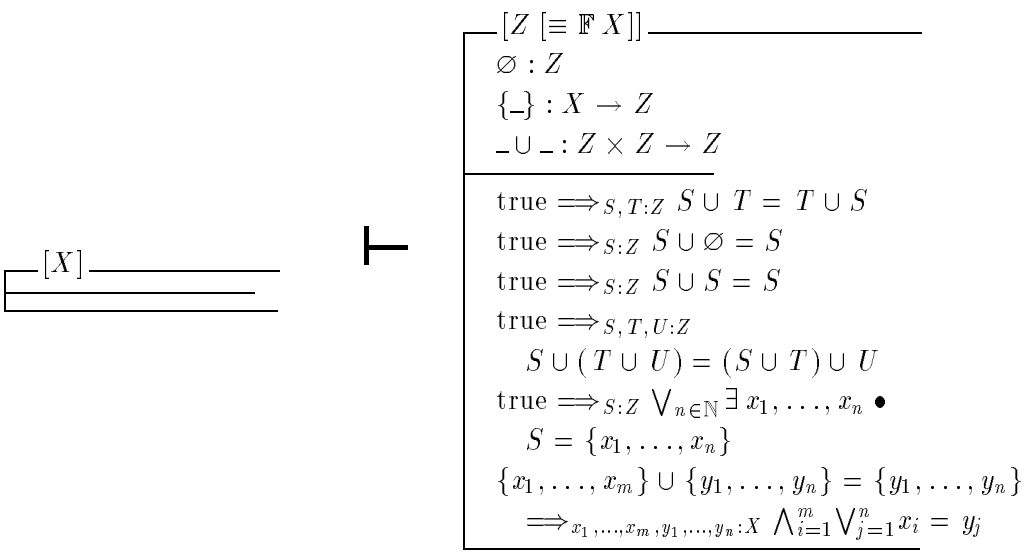
At the level of syntax, there is a certain analogy with type theory. That allows judgements such as "X is a type", "t is a term of type X" "P is a proposition", and "P is true", and its *syntax* provides rules under which in a given context certain syntactic formations yield sound judgements. For instance,

X type, Y type $\vdash X \uplus Y$ type
X type, Y type, $x : X \vdash \iota_1(x) : X \uplus Y$
X type, Y type, $x : X, y : Y, \iota_1(x) = \iota_2(y) \vdash$ false

By grouping together the right-hand sides of such entailments, the schema entailment is able to elevate these syntactic rules into assertions with semantic content.

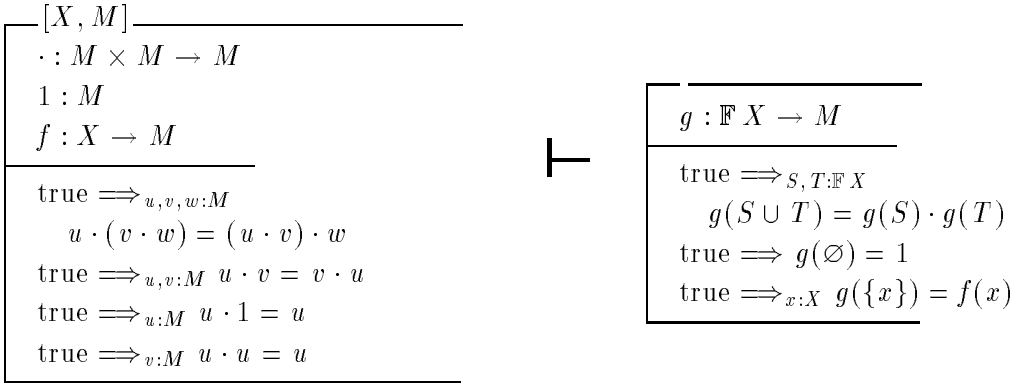
Free Algebras

In [9] finite power sets are characterized in a way that is equivalent to the following definitional extension:



(Obviously from the syntactic point of view, this leaves a lot to be desired because of the infinities. Let us leave aside that issue for the moment.)

A crucial feature of this definition, which can be proved from the schema entailment given above, though it is not at all obvious, is that $\mathbb{F}\,X$ is the free semilattice over X: in other words, we have



In fact, the freeness property is already enough to characterize $\mathbb{F}\,X$ up to isomorphism. Hence the infinitary part of the first entailment, though necessary to ensure a definitional extension, is not needed for subsequent working. Given the two entailments together, we could work just as well if the first had its last two axioms hidden from us.

This phenomenon is rather general. For *any* finitary algebraic theory, the free algebras can be characterized by definitional extensions. However, the details of how this is done are generally boring. They rely on using infinite (though

countable) disjunctions to capture all possible term formations and all possible equations between terms. In practice it suffices to rely on the general theory to say that the definitional extension is possible and go on to the universal property.

Bounded Universal Quantification

Hodges [6] pointed out the specificational significance of universal quantification bounded over finite sets, in the form $\forall\, x \in S \bullet P(x)$. These do not suffer from the observational problems of unbounded (or, rather, type-bounded) universal quantification, for to observe $\forall\, x \in S \bullet P(x)$ where $S = \{x_1, \ldots, x_n\}$, one simply has to observe $P(x_1) \wedge \ldots \wedge P(x_n)$. In fact, such universal quantification is geometric: it can be characterized by a definitional extension (for simplicity we ignore other arguments of P).

$$\frac{[X]\quad P : \mathbb{P}\, X}{} \vdash \frac{}{\forall\, x \in \_ \bullet P(x) : \mathbb{P}\,\mathbb{F}\, X}$$

$$\begin{array}{l} \text{true} \Longrightarrow \forall\, x \in \varnothing \bullet P(x) \\ \forall\, x \in \{x_0\} \bullet P(x) \Longleftrightarrow_{x_0 : X} P(x_0) \\ \forall\, x \in S \cup T \bullet P(x) \Longleftrightarrow_{S,T : \mathbb{F}\, X} \\ \qquad \forall\, x \in S \bullet P(x) \wedge \forall\, x \in T \bullet P(x) \end{array}$$

Proof P is a function from X to Ω, the set of truth values. Since we have to work constructively, there may be truth values other than the classical true and false; nonetheless, Ω has operations corresponding to the connectives of intuitionistic logic, and in particular it's a semilattice under conjunction. Hence by the universal property of $\mathbb{F}\, X$ we get a homomorphism from $\mathbb{F}\, X$ to Ω, and this corresponds to the predicate $\forall\, x \in \_ \bullet P(x) : \mathbb{P}\,\mathbb{F}\, X$.

If we have $\forall\, x \in S \bullet P(x)$ and also $x_0 \in S$ (which means $S = \{x_0\} \cup S$, then we can deduce $P(x_0)$ – an elimination rule for $\forall$. On the other hand, we also have the usual introduction rule in the form

$$\frac{[X]\quad P : \mathbb{P}\, X \quad S : \mathbb{F}\, X \quad x \in S \Longrightarrow_{x : X} P(x)}{} \vdash \frac{}{\text{true} \Longrightarrow \forall\, x \in S \bullet P(x)}$$

Proof Say $S = \{x_1, \ldots, x_n\} = \{x_1\} \cup \ldots \cup \{x_n\}$. Then for each i we have $P(x_i)$, hence $\forall\, x \in \{x_i\} \bullet P(x)$, and it follows that $\forall\, x \in S \bullet P(x)$.

The introduction and elimination rules show that $\forall$ is a true universal quantification.

4 Morphisms of Schemas

Suppose S and T are schemas, and we'd like to describe a transformation from models of S into models of T. One approach is to take a "generic" model of S, one which has no properties at all except that it is a model of S, and show how to construct a model of T from it. Then what could be done for the generic model can also be done for any specific model, thus giving our transformation. This is perfectly common in mathematics. For instance in $x^2 + 7x + 12$, x is a generic number for which any specific number can be substituted; and M is a generic monoid in, "Let M be a monoid. Then M^{op} is a monoid with the same elements and unit, but for which multiplication is defined by $x \cdot_{M^{op}} y = y \cdot_M x$." It also lies behind $\forall$-introduction in logic: to prove $\forall\, x \bullet (P(x) \Rightarrow Q(x))$ you take a generic c satisfying $P(c)$ (i.e. it has no properties except $P(c)$) and prove $Q(c)$ for it.

Classically there is no generic model. For any property ϕ, any classical model must satisfy either ϕ or $\neg\, \phi$ and so (unless one of these is a consequence of S) is not generic – a generic model would *lack both* the properties ϕ and $\neg\, \phi$.

Geometrically, however, we can find a generic model by changing our set theory: we imagine a set theory in which in addition to the ordinary sets we also have "sets" corresponding to the vocabulary of S, and about which nothing is known except that they satisfy the axioms of S. These constitute the generic model of S. We also take everything that can be derived from these by geometric type constructors. This gives a category $\mathcal{S}[S]$ of derived types. Its objects, subobjects and morphisms are the sorts, predicates and functions that can be characterized in definitional extensions of S, and we call it the *G-frame presented by* S. (*G-frame* here is what in Vickers [9] is called a *giraud frame*; but the abbreviation allows us to honour Grothendieck as well. Ordinarily it is called the classifying topos for S, but, as discussed in [9], we reserve the word *topos* for different usage.)

Now to construct a model of T from the generic model of S is to construct a model of T in $\mathcal{S}[S]$. Hence the vocabulary ingredients of T, and the axioms too, must be interpreted as things that can be characterized in definitional extensions of S. By putting them together, we find a single equivalence superschema S' of S into which T can be translated very literally – base types for base types, symbols for symbols, axioms for axioms.

We have now arrived at our definition of *schema morphism* from S to T: an

equivalence superschema S' of S, and a literal translation from T to S'. Apart from some renaming, T can be considered a subschema of S'.

(Experts will understand that this is equivalent to a geometric morphism between classifying toposes: from $[S]$ to $[T]$. However, it is also evidently not the whole story, because we should also define when two morphisms are equivalent, or, more generally, what is a 2-cell (natural transformation) between morphisms.)

A consequence of this discussion is that the task of defining geometric morphisms is reduced to the task of knowing what extensions are definitional.

Note: Maibaum and Veloso [7], who worked on the idea of specification as logical theory, define a morphism from S to T (an implementation of T in S) to be a *conservative* extension S' of S together with a translation from T to S'. Our definitional extensions are on the face of it more restrictive (topos-theoretically, a conservative extension corresponds to a surjective geometric morphism from $[S']$ to $[S]$, while a definitional extension corresponds to an equivalence). However, it appears that not too much has been lost by this, for the increased expressiveness of the infinitary disjunctions allows extensions to be definitional geometrically where they could only be conservative classically. (An example that is presented in more detail in [3] is the theory of natural numbers.) In any case, conservative extensions in geometric logic are not well-behaved (topos-theoretically, surjectivity of geometric morphisms is not preserved by pullback), so we'd prefer not to rely on them.

In order to define composition, we need the following "weakening" property for schema entailment:

Proposition 1 *Let $S \subseteq S'$ be schemas (i.e. S is a subschema of S'), and suppose $S \vdash T$. Then also $S' \vdash T$.* □

Now consider two composable schema morphisms

$$S \xrightarrow{\equiv} S' \longrightarrow T \xrightarrow{\equiv} T' \longrightarrow U$$

Let T', an equivalence superschema of T, be $T + T''$ with $T \vdash T''$. Since S' is (modulo renaming) a superschema of T, we have $S' \vdash T''$, so $S' + T''$ is an equivalence superschema of S' and hence also of S, while it is also a superschema of $T + T''$, *i.e.* T', and hence also of U: we have

$$S \xrightarrow{\equiv} S' \xrightarrow{\equiv} S' + T'' \longrightarrow T' \longrightarrow U$$

This gives the product of the morphisms.

Having defined morphisms, an interesting possibility is that of using universal properties in the category of schemas to characterize theories, instead of using syntactic construction of the schemas. This is a key to modularity in specification, for it considers specifications not by their internal construction

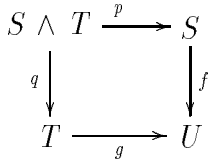
but by their interface, how they relate to all other specifications (a universal property). As a first example, let us prove –

Proposition 2 *Schema conjunction is pullback.*

Proof [sketch only, in view of the 2-categorical complications]

We haven't defined schema conjunction, but it is just as in Z. If S and T are two schemas then $S \wedge T$ has all base types, functions, predicates and axioms of S and T. This is done on the understanding that when a symbol is common to S and T, it appears once only, as itself, in $S \wedge T$ – there is no attempt to make two copies and distinguish between them. This results in a pullback rather than a product.

Let U be the vocabulary common to S and T. U is a subschema of both S and T, and S and T are both subschemas of $S \wedge T$, so we have a square –



We show that it is a pullback. Certainly, because all the morphisms correspond to schema inclusions (no equivalence superschemas needed), we have $p;\ f = q;\ g$. Now suppose we have $p' : V \to S$ and $q' : V \to T$ with $p';\ f = q';\ g$. We have equivalence superschemas V_1 and V_2 of V such that V_1 is a superschema of S and V_2 a superschema of T. By extending V_1 with the extra ingredients used in V_2, we can find an equivalence superschema V' of V that contains both V_1 and V_2. Hence it also includes both S and T, and hence also $S = T$, so we have a morphism $h : V \to S \wedge T$ such that $h;\ p = p'$, $h;\ q = q'$.

Specifying Morphisms

A morphism from S to T says exactly how models of S are to be transformed into models of T. However, one can imagine looser specifications. Consider a schema extension $S + S_1$ of the form

Although we haven't included them, this pattern could easily include "result types" (types calculated from the arguments), auxiliary values used to help

express the postconditions, and changing states (initial state as an argument, final state as a result). Note that the postconditions can happily refer to the arguments.

A translation from T to $S + S_1$ in a sense specifies a morphism from S to T, but doesn't implement one – unless S_1 is definitional over S. To implement the specification is to give a further extension of $S + S_1$ that's definitional over S.

5 Conclusions

The ideas presented here plainly have several radical differences from the conventional view of Z as set out in Brien and Nicholls [1].

To mention first one of the less obvious, we have adopted rather fundamentally the idea of specification as theory presentation ([8]). Hence, whatever you think makes a good semantics of theories (and we have implicitly used categorical logic and classifying toposes) should also be the semantics for specifications. This provides an broad mathematical rationale for the semantic account.

A second difference from the Z standard is that we have rather glossed over syntactic issues – not because we believe them unimportant, but because we feel that good syntactic solutions are still needed for some of the problems of expression. (We have already mentioned the question with schema entailments of how best to have them fulfill two different roles: the assertion of essentially unique existence, and the introduction of notation.) A fair proportion of the Z standard is devoted to the syntactic issues of how the symbols introduced in a schema fit in an environment of symbols already established. Indeed, it is rather difficult there to tease out any meaning of specifications that does not depend on the symbols. Very roughly, I believe the distinction is that in Z names are global, and structures are seen globally as assigning carriers and so on to some of the global names; whereas in GeoZ names and structures are understood locally for a particular schema, and – ideally at least – names in different schemas are related by explicit translations (though we have mentioned schema inclusion, which has implicit translation determined by symbolic identity). Hence the names in GeoZ have less primary importance. Related to this is the treatment of generic types and values, though I must confess to not finding the Z standard account very clear.

Finally, the most striking difference is the use of geometric logic. Since this restricts the language quite considerably – in permitted type constructors and the form of axioms ("schema predicates") – one is justified in asking whether the restrictions are acceptable, notwithstanding our rationale in terms of observations. Ultimately this can only be determined by putting the restricted language into practice, but supporting evidence has come from the work in [6]

"IZ". Independently of the ideas of geometric logic, Hodges has been looking at a similarly restricted logic with no negation or implication, nor universal quantification except when bounded over finite sets. By examining the case studies in [5], he has formed the opinion that the restricted logic suffices.

Even if the restrictions still allow us to write reasonable specifications, one might yet ask what is the benefit of imposing them on ourselves. Again any suggested benefit would have to be evaluated in practice, but our intention is that in GeoZ it should be much harder to write meaningless specifications. For instance, a postcondition for a procedure that says "flag = true iff all crows are black", with its universal quantification over an observationally infinite set, is much less meaningful than one that says "flag = true iff all crows *recorded in the database* are black". In GeoZ we are forced to be aware that a set must be finite if we are to store it in the computer and manipulate it.

Acknowledgements I should like to thank Mark Dawson, not only for the many helpful discussions on how best to turn these ideas into real specification language (some of the fruits of which are in [3]), but also for converting the paper into LaTeX and for his general LaTeXpertise. The ideas have also benefitted from being aired in a discussion group involving Tom Maibaum and Jose Fiadeiro.

The work was assisted financially by the British Science and Engineering Research Council under the "Foundational Structures in Computing Science" research project at Imperial College.

References

[1] S. M. Brien and J. E. Nicholls. Z base standard. Technical Monograph PRG-107, Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, UK, November 1992. Accepted for ISO standardization, ISO/IEC JTC1/SC22.

[2] G.L. Burn, S.J. Gay, and M.D. Ryan, editors. *Theory and Formal Methods 1993: Proceedings of the First Imperial College, Department of Computing, Workshop on Theory and Formal Methods*, Isle of Thorns Conference Centre, Chelwood Gate, Sussex, UK, 29–31 March 1993. Springer-Verlag Workshops in Computer Science.

[3] M. Dawson and S.J. Vickers. Towards a GeoZ toolkit. In Hankin et al. [4].

[4] C.L. Hankin, I.C. Mackie, and R. Nagarajan, editors. *Theory and Formal Methods 1994: Proceedings of the Second Imperial College, Department of Computing, Workshop on Theory and Formal Methods*, Møller Center, Cambridge, UK, 11–14 September 1995. IC-Press.

[5] I.J. Hayes, editor. *Specification Case Studies*. Series in Computer Science. Prentice Hall International, 2nd edition, 1993.

[6] W. Hodges. Another semantics for Z, 1990.

[7] T. Maibaum, P. Veloso, and M. Sadler. Abstact specification theory. Technical report, Department of Computing, Imperial College, 1991.

[8] W. Turski and T. Maibaum. *The Specification of Computer Programs*. Addison-Wesley, 1987.

[9] S. J. Vickers. Geometric logic in computer science. In Burn et al. [2], pages 37–54.

[10] Steven J. Vickers. Geometric theories and databases. In Fourman, Johnstone, and Pitts, editors, *Applications of Categories in Computer Science*, volume 177 of *LMS Lecture Note Series*, pages 288–314. Cambridge University Press, 1992.