TR–??–??

Notes on sketches

Michael Barr*

June 11, 1999

*These notes are based on a series of lectures on sketches given at the ETL in Tsukuba. Parts of it are adapted from [Barr & Wells, 1995]. I would like to acknowledge the support of the COE budget of STA Japan.

1 In the beginning

1.1 Universal algebra In the 1930s, Garrett Birkhoff, Marshall H. Stone and others created the subject known as universal algebra. This was one manifestation of the rise of abstraction in mathematics that started in the mid 19th century. (It is interesting to note that this rise in abstraction coincided, more or less, with increasing abstraction in music, in art, in literature, in science and in other endeavors.) Although universal algebra was not sufficient to describe all kinds of mathematical structures (topological structures were notably absent), it was a vehicle for describing many kinds, notably in algebra. We begin with a brief description.

1.2 Signatures A **theory** in universal algebra began with a **signature**, that is an $\mathbf{N}$-graded set of **operations**, $\mathcal{S} = \{\Omega_0, \Omega_1, \ldots\}$. An element $\omega \in \Omega_n$ is called an n**-ary operation**. When $n = 0, 1, 2, 3, 4$, an n-ary operation is commonly called a **nullary**, **unary**, **binary**, **ternary**, and **quaternary** operation, respectively. A nullary operation is also often called a **constant**. An n-ary operation is also said to have **arity** n.

1.3 Models A **model** of this signature is simply a set S together with a function $\omega: S^n \longrightarrow S$ for each $\omega \in \Omega_n$.

So, for example, you might have a signature $\mathcal{S}$ in which Ω_0, Ω_1, and Ω_2 each had one element and all the others were empty. A model of this signature would be a set S equipped with functions $\omega_0: S^0(= 1) \longrightarrow S$, $\omega_1: S^1(= S) \longrightarrow S$, and $\omega_2: S^2(= S \times S) \longrightarrow S$. It can be thought of as the raw material of a group, although it is not yet, of course, a group.

1.4 homomorphisms of models If $\mathcal{S}$ is a signature and S and T are models of $\mathcal{S}$, that is sets equipped with the operations, then a function $f: S \longrightarrow T$ is called a **homomorphism of models** if for all $n \in \mathbf{N}$, all $\omega \in \Omega_n$, and all $x_1, x_2, \ldots, x_n \in S$ we have

$$f(\omega(x_1, \ldots, x_n)) = \omega(f(x_1), \ldots, f(x_n))$$

1.5 Derived operations and equations Beginning with the operations in a signature, there are certain derived operations, defined inductively as follows. First, for each pair of integers $i \leq n$, there is a derived n-ary operation p_i (or p_i^n) whose value at a model is given by $p_i(x_1, \ldots, x_n) = x_i$. These are called projections and are derived operations even if every Ω_n is empty). It is convenient to also denote p_1^1 by id, since its value on any element is that element. Then if ω is a k-ary operation and $\omega_1, \omega_2, \ldots, \omega_k$ are n-ary operations, there is an n-ary derived operation denoted $\omega \circ (\omega_1, \ldots, \omega_k)$ whose value in a model is given by

$$\omega \circ (\omega_1, \ldots, \omega_k)(x_1, \ldots, x_n) = \omega(\omega_1(x_1, \ldots, x_n), \ldots, \omega_k(x_1, \ldots, x_n))$$

Then an **equation** is the assertion of an equality of two operations, usually derived. For example, in the theory described above, there are two derived ternary operations. If the binary operation is called μ, then there are ternary operations $\mu\circ(\mu\circ(p_1, p_2), p_3)$ and $\mu\circ(p_1, \mu\circ(p_2, p_3))$. To see what these mean, note that $p_1(x, y, z) = x$ and so on. Then we have,

$$\mu\circ(p_1, p_2)(x, y, z) = \mu(p_1(x, y, z), p_2(x, y, z)) = \mu(x, y)$$

so that

$$\mu\circ(\mu\circ(p_1, p_2), p_3)(x, y, z) = \mu(\mu(x, y), z)$$

and similarly,

$$\mu\circ(p_1, \mu\circ(p_2, p_3))(x, y, z) = \mu(x, \mu(y, z))$$

Thus the equality of these two operations on a model is just the associativity of μ in the usual sense. In a similar way, you can write equations that specify that the nullary operation is an identity element for μ and that the unary operation is an inverse for μ.

Then a theory of universal algebra is simply a signature and a set of equations. A model of the theory is a model of the signature that satisifes the equations. That is, the operations on the model satisfy the equations of the theory. A homomorphism between models is the same as a homomorphism between models of the signature.

2 Enter Lawvere

2.1 Theory as category This was the situation in 1963, when F. W. Lawvere burst on the scene with his Ph.D. thesis. It is probably not much of an exaggeration to say that this and much of subsequent career has been devoted to "functorializing" mathematics. He showed how to make an equational theory into a category in such a way that models are functors and homomorphisms of models are natural transformations. This is done as follows.

2.2 The full clone The set of all operations—both given given and derived—is called the **full clone** of operations of the theory. This full clone can be used to build a category. The objects of the category are the natural numbers and an arrow $m \longrightarrow n$ is an n-tuple of m-ary operations of the theory. In particular, when $n = 0$, there is a unique arrow $m \longrightarrow n$ in the theory. If $\omega = (\omega_1, \ldots, \omega_n): m \longrightarrow n$ and $\psi = (\psi_1, \ldots, \psi_m): k \longrightarrow m$, then Lawvere defined $\omega\circ\psi = (\omega_1\circ\psi, \ldots, \omega_n\circ\psi): k \longrightarrow n$. This is readily seen to be a category with the n-tuple $(p_1, \ldots, p_n): n \longrightarrow n$ as identity arrow at n.

Finally, if two operations are to be equal in the theory, then we make them equal in the category. This implies as well that any derived operations that arise by substituting one for the other will also be equal. Thus the equations in the theory give rise to commutative diagrams in the associated category. Let us call this category **Th**.

In this way, the full clone gives rise to a category. Conversely, the category gives back the full clone if we define the n-ary operations to be the arrows from $n \longrightarrow 1$ in the category.

2.3 Lawvere theories The category so defined is characterized by the following properties. Its objects can be thought of as the natural numbers 0, 1, 2, ... Since an arrow $m \longrightarrow 1$ is an n-ary operation and an arrow $m \longrightarrow n$ is an n-tuple of the n-ary operations, this means that $\mathrm{Hom}(m, n) \cong \mathrm{Hom}(m, 1)^n$. This in turn means that in the category **Th**, the object 1 is the n-fold power of the object 1. It might be a little easier if we denoted the objects X^0, X^1, ..., X^n, ..., but this notation is historically correct.

2.4 Models Now suppose that M is a model of the original theory. This means that for each $\omega \in \Omega_m$ there is given a function $M\omega\colon M^m \longrightarrow M$. If $\omega_1, \ldots, \omega_n$ is an n-tuple of m-ary operations, then you get a function $(\omega_1, \ldots, \omega_n)\colon M^m \longrightarrow M^n$ given by the formula

$$(\omega_1, \ldots, \omega_n)(x_1, \ldots, x_m) = (\omega_1(x_1, \ldots, x_m), \ldots, \omega_n(x_1, \ldots, x_m))$$

Thus we can make M into a *functor* on **Th**; define $M(n) = M^n$ and $M(\omega_1, \ldots, \omega_n) = (M\omega_1, \ldots, M\omega_n)$. It is easy to see that M so defined is a functor. Moreover, M preserves products since $M(n) = M(1)^n$. This allowed Lawvere to *define* a **theory** (now usually called a **Lawvere theory**) as a category whose objects are all the finite products of one specified object and a model of the theory as a product preserving functor. One obvious advantage of this is that it makes it natural to define models in other categories, so that a topological group is nothing but a model of the theory of groups in the category of topological spaces. Of course, this had been done before, but that at least made it systematic.

One characterization of a Lawvere theory is as a category **Th** equipped with a contravariant product preserving functor from the category of finite sets that is bijective on objects.

2.5 Generalizations Lawvere's definition was an eye-opener. Suddenly, a theory was nothing but a category and a model was a (product-preserving) functor. This seems (to me) much simpler than the previous definition. The equations of the theory are now simply commutative diagrams and all the methods of category theory can now be brought to bear on the subject. Before doing this, I wanted to mention that there are some obvious generalizations. The first, by F. E. J. Linton [1966, 1969a,b], was to drop the finiteness implicit in the fact that the objects are natural numbers. Instead, we can define a theory to be a category whose objects are sets with the property that the object S is the S-fold power of the object 1. Again, it might be useful to call the objects X^S where S is an arbitrary set. Another way of saying this is to say that there is a contravariant product preserving functor $\mathbf{Set}^{\mathrm{op}} \longrightarrow \mathbf{Th}$ that is bijective on objects. What this means is that there are now allowed to be operations of arbitrary arity. For example, it is not hard to show that there is a theory whose model category is equivalent to the category of commutative C*-algebras. It includes operations that allow infinite absolutely convergent sums to be formed. (Just for the record, the underlying functor used is the unit ball. For any sequence $a_1,\ a_2,\ \ldots$ of complex numbers for which $\sum |a_i| \leq 1$, there is an $\aleph_0$-ary operation $\{x_n\} \mapsto \sum a_i x_i$ which is convergent for a sequence of elements of the unit ball.)

Another direction of generalization is to allow multi-sorted theories. One example of a multi-sorted theory is that of all monoid actions. A monoid action is defined as a pair (M, S) where M is a monoid and S is a set on which M acts. An arrow is a pair $(\varphi, f)\colon (M, S) \longrightarrow (M', S')$ where $\varphi\colon M \longrightarrow M'$ is a monoid homomorphism and $f\colon S \longrightarrow S'$ is a function that satisfies $f(ms) = \varphi(m)f(s)$ for all $m \in M$ and $s \in S$. In an obvious way, there is a theory for this category, but the objects are the products of powers of two given objects, one corresponding to the monoid part and one to the set part. This is an example of a 2-sorted theory, but the number of possible sorts is unlimited.

In some cases, but not in this one, it is possible to replace a 2-sorted (or more) theory by a single-sorted one. Here is the reason it is not possible. In the category of models of a single-sorted theory, for any object A, either $A = \emptyset$ or the obvious diagram $A \times A \rightrightarrows A \longrightarrow A$ is a coequalizer. Now we can take $A = (M, \emptyset)$ for which $A \times A \rightrightarrows A \longrightarrow (1, \emptyset)$ is a coequalizer and $(1, 0)$ is not terminal, since $(1, 1)$ is. Nor is $(M, \emptyset)$ empty since there is at most empty model and M is arbitrary.

Finally, we can define a **product theory** as a category with products. A model is a product preserving functor into sets. A homomorphism between models is a natural transformation between functors. This definition pushes the Lawvere theory about as far as it can go without changing its character in a fundamental way.

Later generalizations included limit theories which are categories with limits and models are limit-preserving functors. This will be discussed in further detail when we come to sketches.

3 Ehresmann and sketches

Charles Ehresmann was a French differential geometer who got interested in category theory late in his career. He founded a school on the subject that is still active. Although I am guessing to some extent in this history, I imagine he must have learned of Lawvere's work and reasoned while Lawvere had captured the full clone of a theory, but the original idea à la Birkhoff, was much smaller, usually finite, at least for the common theories we usually use, while Lawvere's theories were always infinite (with two trivial exceptions). At any rate, Ehresmann (often in conjunction with Andrée Bastiani, later Bastiani-Ehresmann), wrote a series of papers defining and working out many of the important properties of **sketches** (French *esquisses*). See [Ehresmann, 1968, 1969], [Bastiani & Ehresmann, 1972] and [Bastiani, 1973]. He gave a number of different definitions and variations, so that he really defined the idea of a sketch. Thus Wells and I felt justified in giving further variations as seemed appropriate. In particular, most (but not all) of Ehresmann's definitions were based on categories, while we felt that this already put in more structure than was strictly necessary. Without going into further detail, I will describe the definition as appears in [Barr & Wells, 1985].

Ehresmann's students have continued his work; however they virtually always publish in the pseudo-journal *Diagrammes* which is hardly ever distributed outside of France. As a result, they spend the rest of their time engaging in nasty priority disputes with the rest of the world, going so far as to make accusations of plagiarism against those who could have had no opportunity to see their work. For example, the seminal work of Makkai and Paré included the proof that sketchable is the same as accessible. This was basically anticipated by the Ehresmann student C. Lair. I cannot give citations as I have seen it only once and there would be no point since you have no access to it. Anyway, see [Makkai & Paré, 1990]. Let me make it clear that I am not impugning the quality of their work, only their failure to adequately document it.

3.1 Graphs The very simplest kind of sketch is a **graph**. By a graph, I mean a directed multigraph that is allowed to contain loops. Any category can be thought of as a graph by just forgetting the composition. If $\mathcal{G}$ is a such a graph, a graph morphism into a category $\mathcal{C}$ is just a graph morphism into that underlying graph. This means it takes nodes to objects, arrows to arrows, and preserves domain and codomain.

A model (in **Set**) of this simplest kind of sketch amounts to prescribing a set for each node in the graph and a function between sets for each arrow in the graph. This is what might be called a free multi-sorted linear theory (free because there are no equations and linear because all operations are unary). Neither nullary, binary nor any higher arity operation can be prescribed with this simplest kind of sketch, nor are equations possible. The only familiar theory that can be described with this kind of sketch is that of M-sets, where M and, even so, M must be free. For example, $\mathbf{N}$ is a free monoid on one generator, so the category of $\mathbf{N}$-sets is a model of this kind of sketch. The graph has one node and one loop. An $\mathbf{N}$-set is just a set equipped with an endomorphism.

3.2 Commutative diagrams When the target graph of a diagram is the underlying graph of a category some new possibilities arise, in particular the concept of

commutative diagram.

In this situation, we will not distinguish in notation between the category and its underlying graph: if $\mathcal{I}$ is a graph and $\mathcal{C}$ is a category we will call a graph morphism $D{:}\,\mathcal{I} \longrightarrow \mathcal{C}$ a **diagram** in $\mathcal{C}$ with **shape** $\mathcal{I}$. We will refer to $\mathcal{I}$ as the **shape graph** of D.

We say that D is **commutative** (or **commutes**) provided for any nodes i and j of $\mathcal{I}$ and any two paths

$$\begin{tikzcd}
 & k_1 \arrow[r, "s_2"] & k_2 \arrow[r] & \cdots \arrow[r] & k_{n-2} \arrow[r, "s_{n-1}"] & k_{n-1} \arrow[dr, "s_n"] & \\
i \arrow[ur, "s_1"] \arrow[dr, "t_1"'] & & & & & & j \\
 & l_1 \arrow[r, "t_2"'] & l_2 \arrow[r] & \cdots \arrow[r] & l_{m-2} \arrow[r, "t_{m-1}"'] & l_{m-1} \arrow[ur, "t_m"'] &
\end{tikzcd}$$

from i to j in $\mathcal{I}$, the two paths

$$\begin{tikzcd}
 & Dk_1 \arrow[r, "Ds_2"] & Dk_2 \arrow[r] & \cdots \arrow[r] & Dk_{n-2} \arrow[r, "Ds_{n-1}"] & Dk_{n-1} \arrow[dr, "Ds_n"] & \\
Di \arrow[ur, "Ds_1"] \arrow[dr, "Dt_1"'] & & & & & & Dj \\
 & Dl_1 \arrow[r, "Dt_2"'] & Dl_2 \arrow[r] & \cdots \arrow[r] & Dl_{m-2} \arrow[r, "Dt_{m-1}"'] & Dl_{m-1} \arrow[ur, "Dt_m"'] &
\end{tikzcd} \qquad (*)$$

compose to the same arrow in $\mathcal{C}$. This means that

$$Ds_n \circ Ds_{n-1} \circ \ldots \circ Ds_1 = Dt_m \circ Dt_{m-1} \circ \ldots \circ Dt_1$$

3.3 Much ado about nothing There is one subtlety to the definition of commutative diagram: what happens if one of the numbers m or n in Diagram $(*)$ should happen to be 0? If, say, $m = 0$, then we interpret the above equation to be meaningful only if the nodes i and j are the same (you go nowhere on an empty path) and the meaning in this case is that

$$Ds_n \circ Ds_{n-1} \circ \ldots \circ Ds_1 = \mathrm{id}_{D_i}$$

(you do nothing on an empty path). In particular, a diagram D based on the graph

$$\begin{tikzcd}
\mathrm{i} \arrow[loop above, "\mathrm{e}"]
\end{tikzcd}$$

commutes if and only if $D(e)$ is the identity arrow from $D(i)$ to $D(i)$.

Note, and note well, that both shape graphs

$$\begin{tikzcd} i \arrow[loop above, "e"] \end{tikzcd} \qquad\qquad \begin{tikzcd} i \arrow[r, "d"] & j \end{tikzcd}$$

(a) (b)

have models that one might think to represent by the diagram

$$\begin{tikzcd} A \arrow[loop above, "f"] \end{tikzcd}$$

but the diagram based on (a) commutes if and only if $f = \mathrm{id}_A$, while the diagram based on (b) commutes automatically (no two nodes have more than one path between them so the commutativity condition is vacuous).

We will always picture diagrams so that distinct nodes of the shape graph are represented by distinct (but possibly identically labeled) nodes in the picture. Thus a diagram based on (b) in which d goes to f and i and j both go to A will be pictured as

$$A \xrightarrow{f} A$$

In consequence, one can always deduce the shape graph of a diagram from the way it is pictured, except of course for the actual names of the nodes and arrows of the shape graph.

3.4 Graphs with diagrams The next simplest kind of sketch consists of a graph with diagrams. A graph with diagrams is a pair $\mathcal{S} = (\mathcal{G}, \mathcal{D})$ where $\mathcal{G}$ is a graph and $\mathcal{D}$ is a set of diagrams (with various index graphs).

It does not make any sense to ask that a diagram in a graph commute, but it does make sense in a category. Thus if $\mathcal{S} = (\mathcal{G}, \mathcal{D})$ is a graph with diagrams, we can define a model of $\mathcal{S}$ to be a model M of $\mathcal{G}$ such that for any $D: \mathcal{I} \longrightarrow \mathcal{G}$ in $\mathcal{D}$, the composite diagram $M \circ D$ commutes. As in a Lawvere theory, the commutation of diagrams is a way of putting equations into the theory. For example, using a graph with diagrams we can make a sketch whose models are the category of M-sets for an arbitrary monoid M. The sketch will have one node and have an arrow for each generator of the monoid. Whenever there is a relation among generators, say $x_1 x_2 \cdots x_n = y_1 y_2 \cdots y_m$ we put in a diagram as based on the graph $\mathcal{I}$ above with $Ds_i = x_{n-i}$ and $Dt_j = y_{m-j}$. There is some complication when one of the indices is 0. Suppose, for example, $m = 0$. We cannot simply take $i = j$ for that will result in many more loops than we want. The way to do is to replace the equation by two diagrams, one saying that $x_1 x_2 \ldots x_n = e$ and the second saying that $e = 1$.

This kind of sketch is still linear and has very little expressive power since the operations are limited to functions of one variable. Here is an example.

3.5 Example We now consider an example of a linear sketch which has diagrams. Suppose we wanted to consider sets with permutations as structures. This would be a u-structure (S, u) with u a bijection. We can force u to go to a bijection in **Set**-models by requiring that it have an inverse. Thus the sketch $\mathcal{P}$ of sets with permutations has as graph the graph $\mathcal{G}$ with one node e and two arrows u and v, together with this diagram D:

$$e \underset{v}{\overset{u}{\rightleftarrows}} e \tag{1}$$

based on the shape graph

$$i \underset{y}{\overset{x}{\rightleftarrows}} j \tag{2}$$

A model M of this sketch in **Set** must have $M(e)$ a set, $M(u)$ and $M(v)$ functions from $M(e)$ to itself (since $D(i) = D(j) = e$), and because Diagram (1) must go to a commutative diagram, it must have

$$M(u) \circ M(v) = M(v) \circ M(u) = \mathrm{id}_{M(e)}$$

This says that $M(u)$ and M(v) are inverses to each other, so that they are permutations. Note that a model in any category is an object of that category together with an isomorphism of the object with itself and the inverse of that isomorphism.

3.6 Comparison with Ehresmann At this point, it becomes possible to discuss how the sketches described here compare with Ehresmann's notion. For the most part, Ehresmann supposed that his sketches were based on categories, not graphs. It is not hard to see that a category is actually equivalent, in the sense of having the same models, to a graph with diagrams. For any category has an underlying graph, in which you forget the composition. If you take for diagrams all the commutative triangles (that is to say, all the instances of composition) and all the identity arrows (that is to say all instances of identity) you get a graph with diagrams whose models in any other category are obviously the functors. To go the other way, beginning with any graph, there is the path category in which you formally compose any two arrows whose domain and codomain match, including the empty paths that act as identities. Composition is obvious and gives a category. The diagrams generate a congruence, the least equivalence relation that makes all the diagrams commute and is compatible with composition. The result is a category whose functors are just the models of the original graph.

In fact, in at least one paper (which, unfortunately, I cannot find), Ehresmann and Bastiani used a concept that is even more closely related to my sketches. They defined something whose name I have forgotten, something like quasi-category, which is a graph in which some subset of the composable paths have a given composition. For example, assuming that $\mathrm{dom}(f) = \mathrm{cod}(g)$ and $\mathrm{dom}(g) = \mathrm{cod}(h)$, we might have

$f \circ g \circ h$ defined without, however, supposing that either $f \circ g$ or $g \circ h$ is defined. On the other hand, if, say, both $f \circ g$ and $(f \circ g) \circ h$ are defined, then it is assumed that $(f \circ g) \circ h = f \circ g \circ h$. The notion seems rather awkward, but it is closely related to the idea of a sketch with diagrams.

4 Product sketches

A diagram in a category is called a cone if it has a vertex, a node that has a unique arrow to all other nodes and none to itself. If these are the only arrows in the graph, it is called a discrete cone. If there are only finitely many nodes and arrows, it is called a finite discrete cone. If $\mathcal{I}$ is a category, a cone $L\colon \mathcal{I} \longrightarrow \mathcal{C}$ is called a **limit cone** if it is a limit in the usual sense.

> What is the usual sense? There is an obvious notion of morphism of diagrams that is essentially the same as natural transformation, since nowhere is composition in the domain category used in the definition of natural transformation. The base of a cone is the diagram without the vertex (which is evidently unique, being the target of no arrow). Among all diagrams with the same base, the limit diagram is the final one. We say "the" limit since although not unique it is unique up to a unique isomorphism.

A **product sketch** $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{L})$, where $\mathcal{G}$ is a graph, $\mathcal{D}$ is a set of diagrams in $\mathcal{G}$ and $\mathcal{L}$ is a set of discrete cones in $\mathcal{G}$. A model of $\mathcal{S}$ is a model of $(\mathcal{G}, \mathcal{D})$ that has the additional property of taking every cone in $\mathcal{L}$ to a limit cone. A discrete limit cone is just a product cone, so these are really products. The product sketch is called finite if everything in sight is: the graph is finite; there are finitely many diagrams; the index graph of each one is finite, meaning it has a finite number of nodes and arrows; the number of cones is finite; and each one is based on a finite graph.

Already finite product sketches have a good deal of expressive power. We will illustrate this by describing a sketch whose models are semigroups. We begin with a graph that has three nodes, which we name s, $s \times s$ and $s \times s \times s$. It should be emphasized that these names, while suggestive, have no actual significance. A diagram in a graph can neither commute nor be a limit. Similar remarks apply to the arrows that we are going to label as though they were product projections.

The arrows in the graph are:

$$\begin{tikzcd} & s \times s \arrow[dl, "\pi_1"'] \arrow[d, "\kappa"] \arrow[dr, "\pi_2"] & \\ s & s & s \end{tikzcd}$$

(a)

$$\begin{tikzcd} & s \times s \times s \arrow[dl, "\pi_1"'] \arrow[d, "\pi_2"] \arrow[dr, "\pi_3"] & \\ s & s & s \end{tikzcd}$$

(b)

$$\begin{tikzcd}[column sep=small]
& s \times s \times s \arrow[dl, "\langle \pi_1, \pi_2 \rangle"'] \arrow[dr, "\langle \pi_2, \pi_3 \rangle"] & \\
s \times s & & s \times s
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& s \times s \times s \arrow[dl, "s \times \kappa"'] \arrow[dr, "\kappa \times s"] & \\
s \times s & & s \times s
\end{tikzcd}$$

(c) (d)

We have given the graph in four pictures, but it is just one graph with three nodes and ten arrows. Note that there are two arrows labeled π_1 and two labeled π_2. This is overloaded terminology, used by long convention. For example, the arrows named π_1 and π_2 are necessarily different because their domains and codomains are different.

4.1 Before giving the details of the diagrams and cones, we pause to explain the intent of this example. Up till now, we have described a graph. What is a model of this graph in the category **Set**? We need sets we call $S = M(s)$, $S^2 = M(s \times s)$ and $S^3 = M(s \times s \times s)$. For the moment, the exponents are simply superscripts. In addition we require functions $M(\pi_i): S^2 \longrightarrow S$, $i = 1,\ 2$, $M(\pi_i): S^3 \longrightarrow S$, $i = 1,\ 2,\ 3$, $M(\kappa): S^2 \longrightarrow S$ and $M(s \times \kappa)$ and $M(\kappa \times s)$ from S^3 to S^2. So far, this is nothing familiar, but if we now suppose, as suggested by the notation, that S^2 and S^3 are actually the cartesian square and cube of S and if we make certain subsidiary assumptions given by diagrams to be described later, these data cause S to be a semigroup whose multiplication map is given by $M(\kappa): S \times S \longrightarrow S$.

The subsidiary hypotheses are

(i) The various $M(\pi_i)$ are indeed the projections suggested by the notation.

(ii) $M(\langle \pi_1, \pi_2 \rangle) = \langle M(\pi_1), M(\pi_2) \rangle$ and similarly for $\langle \pi_2, \pi_3 \rangle$.

(iii) $M(\kappa \times s): S \times S \times S \longrightarrow S \times S$ is the unique function (guaranteed by the specification for products) for which the diagram

$$\begin{tikzcd}
S \times S \arrow[d, "M(\kappa)"'] & S \times S \times S \arrow[l, "M\langle \pi_1, \pi_2 \rangle"'] \arrow[d, "M(\kappa \times s)"'] \arrow[dr, "M(\pi_3)"] & \\
S & S \times S \arrow[l, "M(\pi_1)"] \arrow[r, "M(\pi_2)"'] & S
\end{tikzcd}$$

commutes. This diagram merely expresses the fact that $M(\kappa \times s) = M(\kappa) \times \mathrm{id}_S$, which we need to get associativity (in Diagram (d) below).

(iv) There is a similar diagram to express the fact that $M(s \times \kappa) = \mathrm{id}_S \times M(\kappa)$.

(v) Finally, if we want a semigroup, we must express the associative law of the multiplication. This is done by saying that the diagram

$$\begin{tikzcd}
S\times S\times S \arrow[r, "M(s\times\kappa)"] \arrow[d, "M(\kappa\times s)"'] & S\times S \arrow[d, "M(\kappa)"] \\
S\times S \arrow[r, "M(\kappa)"'] & S
\end{tikzcd}$$

commutes.

4.2 Our task will be to express these requirements in our sketch. This is done as follows. We let $\mathcal{D}$ consist of the following diagrams:

$$\begin{tikzcd}
& s\times s\times s \arrow[dl, "\pi_1"'] \arrow[d, "\langle\pi_1,\pi_2\rangle"] \arrow[dr, "\pi_2"] & \\
s & s\times s \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & s
\end{tikzcd}$$

(a)

$$\begin{tikzcd}
& s\times s\times s \arrow[dl, "\pi_2"'] \arrow[d, "\langle\pi_2,\pi_3\rangle"] \arrow[dr, "\pi_3"] & \\
s & s\times s \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & s
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
s\times s \arrow[d, "\kappa"'] & s\times s\times s \arrow[l, "\langle\pi_1,\pi_2\rangle"'] \arrow[d, "\kappa\times s"'] \arrow[dr, "\pi_3"] & \\
s & s\times s \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & s
\end{tikzcd}$$

(c)

$$\begin{tikzcd}
& s\times s\times s \arrow[r, "\langle\pi_2,\pi_3\rangle"] \arrow[dl, "\pi_1"'] \arrow[d, "s\times\kappa"] & s\times s \arrow[d, "\kappa"] \\
s & s\times s \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & s
\end{tikzcd}$$

(d)

$$\begin{tikzcd}
s\times s\times s \arrow[r, "s\times\kappa"] \arrow[d, "\kappa\times s"'] & s\times s \arrow[d, "\kappa"] \\
s\times s \arrow[r, "\kappa"'] & s
\end{tikzcd}$$

(e)

These five diagrams have (e) as their main statement; (c) and (d) are needed to define arrows which occur in (e), and (a) and (b) are needed to define arrows which occur in (c) and (d). This construction is reminiscent of the way you construct progressively higher level procedures in a programming language, culminating in the procedure which actually does what you want.

4.3 The set $\mathcal{L}$ of cones consists of the following:

$$
\begin{tikzcd}
 & s \times s \arrow[dl, "\pi_1"'] \arrow[dr, "\pi_2"] & \\
s & & s
\end{tikzcd}
\qquad
\begin{tikzcd}
 & s \times s \times s \arrow[dl, "\pi_1"'] \arrow[d, "\pi_2"] \arrow[dr, "\pi_3"] & \\
s & s & s
\end{tikzcd}
$$

(a) (b)

Then we say that the model M of $\mathcal{S}$ is a model of the sketch if all the diagrams in $\mathcal{D}$ become commutative diagrams when M is applied and if all the cones in $\mathcal{L}$ become product cones.

With some more arrows, cones and diagrams, this sketch can be progressively modified to become a sketch for monoids, then groups, then commutative groups and then rings. I will introduce some notational conventions that make it easier.

It is not hard to see that product sketches have exactly the expressive of equational theories, at least if the graph is allowed to be large. The Lawvere theories correspond to a special (and not very natural) class of finite sketches that have cones that force each object in a model to be a finite product of a single one.

4.4 The sketch for natural numbers Just about the simplest product sketch is the one for natural numbers. It has two sorts, called 1 and n, two arrows $\zeta: 1 \longrightarrow n$ and $\sigma: n \longrightarrow n$, and just one cone with 1 as its vertex and empty base. A model M of this sketch consists of a set $N = M(n)$ with a chosen element $z = M(\zeta) \in N$ and an endomorphism $s = M(\sigma): N \longrightarrow N$. Such data do not constitute a set of natural numbers (for example, s could be the identity map, or a constant), but rather it is the **initial model**, to be described later, that is the usual natural numbers.

4.5 Sketches for arrays and records It is not hard to give sketches that describe arrays and records in some programming language, such as Pascal. For the former, we take two sorts d and a and for an integer m (the size of the array), we take the cone, along with its associated elements as the arrows

$$
\begin{tikzcd}
 & & a \arrow[dll, "\pi_1"'] \arrow[d, "\pi_i"] \arrow[drr, "\pi_m"] & & \\
d & \cdots & d & \cdots & d
\end{tikzcd}
$$

The idea is that this is a sketch for m-ary arrays of datatype d. It is just the bones of such a sketch; to put flesh on it, you have to also sketch the datatype. For example, suppose we add to this sketch, the sketch for natural numbers just described, together with one cone:

$$\begin{tikzcd} n \arrow[d] \\ d \end{tikzcd}$$

A cone of this shape in a category is a limit cone iff the arrow is an isomorphism. So putting in this cone has the effect of identifying d with n. Of course, we could have given them the same identifier, but it is more systematic to do it this way.

The sketch for records is quite similar. Now we have sorts d_1, d_2, …, d_k and a sort r. There is one cone:

$$\begin{tikzcd} & & r \arrow[dll, "\pi_1"'] \arrow[d, "\pi_i"] \arrow[drr, "\pi_m"] & & \\ d_1 & \cdots & d_i & \cdots & d_m \end{tikzcd}$$

Again, this is only the bones of a sketch and the d_i have to be given additional structure in order to acquire meat.

Many other data structures do not have product sketches because they have underflow conditions for which sum sketches will be required. (There is another point; all real-world datatypes also have overflow conditions, which can also be represented by sum theories. Whether you want to build that in to your sketch involves a trade-off of realism vs. analyzability.)

5 Limit sketches

Limit sketches are just like product sketches except that the cones are no longer required to be discrete. So a limit sketch takes the form $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{L})$ where $\mathcal{G}$ is a graph, $\mathcal{D}$ is a set of diagrams, and $\mathcal{L}$ is a set of cones.

Here is a limit sketch for categories. First I will show that the category of categories is not the category of models of any equational theory. An equational theory, even multi-sorted, has the property that any coequalizer is surjective on each sort and conversely. To see this, consider a typical operation $\omega: s_1 \times s_2 \times \cdots s_n \longrightarrow s$. We pretend it is finitary, but modulo the axiom of choice, nothing depends on that. Now suppose that $f: M \longrightarrow N$ is a morphism of models. If it is not surjective, define a new model K by letting Ks be the image of $Ms \longrightarrow Ns$. Given an operation as above, we have the diagram

$$\begin{tikzcd}
Ms_1 \times \cdots \times Ms_n \arrow[r, two heads] \arrow[d, "M\omega"'] & Ks_1 \times \cdots \times Ks_n \arrow[d, tail] \\
Ms \arrow[d, two heads] & Ns_1 \times \cdots \times Ns_n \arrow[d, "N\omega"] \\
Ks \arrow[r, tail] & Ns
\end{tikzcd}$$

whose diagonal fill-in gives an arrow $Ks_1 \times Ks_2 \times \cdots \times Ks_n \longrightarrow Ks$ that gives the value of the operation at K. The commutativity of each of the squares when the diagonal is filled in shows that both $M \longrightarrow K$ and $K \longrightarrow N$ are morphisms. One shows even more easily that any quotient (or any subobject) of a model of the operations satisfies all the equations that they do. Now a regular epi cannot be factored in such a way that the second factor is monic, so any regular epi must already be surjective. Conversely, if $f\colon M \longrightarrow N$ is surjective, then on each set it is the coequalizer of its kernel pair. In other words, on each node s, we have that

$$Ms \times_{Ns} Ms \rightrightarrows Ms \xrightarrow{fs} Ns$$

is a coeqalizer. But then, given any map $g\colon M \longrightarrow K$ that has equal composite with $M \times_N M \longrightarrow M$, we get, for each node s an arrow $hs\colon Ns \longrightarrow Ks$ such that

$$\begin{tikzcd}
Ms \arrow[r, "fs"] \arrow[d, "gs"'] & Ns \arrow[dl, "hs"] \\
Ks &
\end{tikzcd}$$

commutes for each s. Now for an operation $\omega\colon s_1 \times \cdots \times s_n \longrightarrow s$, we have the diagram

$$
\begin{tikzcd}
Ms_1 \times \cdots \times Ms_n \arrow[rrrr, two heads, "fs_1 \times \cdots \times fs_n"] \arrow[dr, "M\omega"] \arrow[dddrr, "gs_1 \times \cdots \times gs_n"'] & & & & Ns_1 \times \cdots \times Ns_n \arrow[dl, "N\omega"'] \arrow[dddll, "hs_1 \times \cdots \times hs_n"] \\
& Ms \arrow[rr, two heads, "fs"] \arrow[dr, "gs"'] & & Ns \arrow[dl, "hs"] & \\
& & Ks & & \\
& & Ks_1 \times \cdots \times Ks_n \arrow[u, "K\omega"] & &
\end{tikzcd}
$$

The upper and southwest squares commute because f and g are morphisms and the inner and outer triangle commute by definition of h. An easy diagram chase shows that the southeast square commutes when preceded by $fs_1 \times \cdots \times fs_n$, which is epic and can be cancelled. Thus we conclude that h is a homomorphism. It follows that the various underlying set functors (one for each sort) preserve regular epis. Since they are well known to preserve all limits and since a pullback in Set of a regular epimorphism is a regular epimorphism, it further follows that a pullback in any equational category of a regular epi is regular epic. (Categories that satisfy that condition are called **regular**). Thus to show that a category is not equational, it suffices to show it is not regular.

But being regular is not sufficient. The category of torsion-free abelian groups is regular, but not equational. The reason in that case is that there is an equivalence relation that is not effective—the kernel pair of some arrow—not possible in an equational category. For example, the equivalence relation on $\mathbf{Z}$ of being congruent modulo 2 is not the kernel pair of any arrow to a torsion-free group. Regularity, effective equivalence relations, and the existence of a projective generating set are necessary and sufficient in order that a cocomplete category be equational. The set of projective generators can be taken as the set of sorts.

Now for **Cat**, we simply observe that the coequalizer of the two maps $\mathbf{1} \rightrightarrows \mathbf{2}$ is the category we call $\mathbf{N}$ that has a single object and the natural numbers as the monoid of endomorphisms of that object. The reason is that if we begin with the single arrow $f\colon \cdot \longrightarrow \cdot$ and identify the domain and codomain of f, we are now able to form the composites $f \circ f$, $f \circ f \circ f$, $\ldots$, and these are all different. The reason they are all different is that we have a sequence

$$\mathbf{1} \rightrightarrows \mathbf{2} \longrightarrow \mathbf{N}$$

and therefore that coequalizer, whatever it is, has to be able to have a functor to $\mathbf{N}$, which is possible only if there are no equations. Thus the arrow $\mathbf{2} \longrightarrow \mathbf{N}$ is a regular epimorphism. Now it certainly does not look surjective, but we have to eliminate the possibility that there is some peculiar underlying functor that makes it so. Instead, we will show that it is not pullback stable. Let $2\mathbf{N}$ denote the subcategory of $\mathbf{N}$ that consists of only the even powers of f. It is evident that

$$\begin{tikzcd} 2 \arrow[r] \arrow[d] & 2\mathbf{N} \arrow[d] \\ \mathbf{2} \arrow[r] & \mathbf{N} \end{tikzcd}$$

is a pullback (2 is the discrete category with 2 objects) and that the arrow $2 \longrightarrow 2\mathbf{N}$ is *not* a regular epi. In fact a quotient of a discrete category is discrete.

The crucial thing about a category is that composition is only partially defined. The good thing is that you can explicitly describe the domain of the partial operation, the set of composable pairs, in terms of operations that are globally defined. This led Peter Freyd to define the notion of an **essentially algebraic theory** (sometimes "algebraic" is used as a synonym for "equational") as one with total and partial operations, stratified in such a way that the tier 0 operations are globally defined and the domains of tier n operations can be defined equationally in terms of operations of tiers $n-1$ and lower. One could make up artificial examples, but there does not seem to be any real need to go beyond tier 1 and, in any case, limit theories encompass them all, while making the stratification unnecessary.

A sketch for categories could be made with four sorts, o, a, $a \times_o a$, and $a \times_o a \times_o a$. The notation suggests that these should be viewed as standing for objects, arrows, composable pairs of arrows, and composable triples of arrows, respectively. There are arrows $\gamma, \delta\colon a \longrightarrow o$, $\iota\colon o \longrightarrow a$, $\kappa\colon a \times_o a \longrightarrow a$, as well as various "housekeeping" arrows, such as π_1, π_2, $\pi_3\colon a \times_o a \times_o a \longrightarrow a$, $\kappa \times_o a\colon a \times_o a \times_o a \longrightarrow a \times_o a$, and so on. We also require an arrow we will call $\gamma\iota$, whose purpose will become apparent. There are diagrams There are a number of routine diagrams, but the most important are

$$\begin{tikzcd} a \times_o a \times_o a \arrow[r, "\kappa \times a"] \arrow[d, "a \times \kappa"'] & a \times_o a \arrow[d, "\kappa"] \\ a \times_o a \arrow[r, "\kappa"'] & a \end{tikzcd}$$

$$\begin{tikzcd} o \arrow[r, "\iota"] \arrow[rr, bend left, "\gamma\iota"] & a \arrow[r, shift left, "\gamma"] \arrow[r, shift right, "\delta"'] & o \end{tikzcd}$$

There are two cones:

$$\begin{tikzcd}
& a \times_o a \arrow[dl, "p_1"'] \arrow[d] \arrow[dr, "p_2"] & \\
a \arrow[r, "\delta"'] & o & a \arrow[l, "\gamma"]
\end{tikzcd}$$

$$\begin{tikzcd}
& & a \times_o a \times_o a \arrow[dll, "p_1"'] \arrow[dl] \arrow[d, "p_2"] \arrow[dr] \arrow[drr, "p_3"] & & \\
a \arrow[r, "\delta"'] & o & a \arrow[l, "\gamma"] \arrow[r, "\delta"'] & o & a \arrow[l, "\gamma"]
\end{tikzcd}$$

Since the case of torsion free abelian groups was mentioned, it is interesting to see what a sketch for that theory might look like. Begin with a sketch for abelian groups that has a node a representing the group and an arrow α representing the addition. Then for each positive integer n, let there be a node a_n and for each non-negative integer, let there be an arrow $\mu_n: a \longrightarrow a$, an arrow $\theta_n: a_n \longrightarrow a$, and an arrow $\varphi_n: a_n \longrightarrow a_1$. finally, we arrow $a \times \mu_n: a \times a \longrightarrow a \times a$, and $\langle \mathrm{id}, \mathrm{id} \rangle: a \longrightarrow a \times a$. We need diagrams to make $\langle \mathrm{id}, \mathrm{id} \rangle$ and $a \times \mu_n$ be what they appear to be and we need a diagram

$$\begin{tikzcd}
a \arrow[r, "\langle \mathrm{id} \times \mathrm{id} \rangle"] \arrow[d, "\mu_n"'] & a \times a \arrow[d, "a \times \mu_n"] \\
a & a \times a \arrow[l, "\mu"]
\end{tikzcd}$$

and, the crucial diagram

$$\begin{tikzcd}
& a_n \arrow[dl, "\varphi_n"'] \arrow[dr, "\theta_n"] & \\
a_1 \arrow[rr, "\theta_1"'] & & a
\end{tikzcd} \qquad (*)$$

For each positive integer n, let there be a cone

$$\begin{tikzcd}
& a_n \arrow[dl, "\theta_n"'] \arrow[dr] & \\
a \arrow[rr, shift left, "\mu_n"] \arrow[rr, shift right, "\mu_0"'] & & a
\end{tikzcd}$$

The interpretation of all this is that θ_n is interpreted as the inclusion of the equalizer of multiplication by n and 0. Thus it is the set of elements annihilated by n. The existence of φ_n and the commutation of $(*)$ means that any elemented annihilated by n is already annihilated by 1, that is, is 0.

5.1 A sketch for binary trees Here we include a sketch for binary trees. The problem with binary trees is that while you may normally follow the left or right branch, there is also an empty tree. One way of dealing with this will be given after we introduce sum sketches. It is also possible by introducing a boolean test. If you actually wanted to implement this sketch, it is not clear that the implementations of the two would differ much, if at all.

We have the following basic nodes in the sketch: $1, t,\ t^+,\ b,\ n$. These should be thought of as representing the types of binary trees, nonempty binary trees, the Boolean algebra 2 and the natural numbers, respectively. We have the following operations:

$\mathsf{empty}\colon 1 \longrightarrow t$	$\mathsf{empty?}\colon t \longrightarrow b$	$\mathsf{incl}\colon t^+ \longrightarrow t$
$\mathsf{val}\colon t^+ \longrightarrow n$	$\mathsf{left}\colon t^+ \longrightarrow t$	$\mathsf{right}\colon t^+ \longrightarrow t$
$\mathsf{zero}\colon 1 \longrightarrow n$	$\mathsf{succ}\colon n \longrightarrow n$	$\mathsf{true}\colon 1 \longrightarrow b$
$\mathsf{and}\colon b \times b \longrightarrow b$	$\mathsf{not}\colon b \longrightarrow b$	

The intended meaning of these operations is as follows: the constant $\mathsf{empty}\langle\rangle$ is the empty tree; empty? is the test for whether a tree is the empty tree; incl is the inclusion of the set of nonempty trees in the set of trees; $\mathsf{val}(T)$ is the datum stored at the root of the nonempty tree T; $\mathsf{left}(T)$ and $\mathsf{right}(T)$ are the right and left branches (possibly empty) of the nonempty tree T, respectively. The remaining operations are the standard operations appropriate to the natural numbers and the Boolean algebra 2.

We require that

$$\begin{tikzcd} & t^+ \arrow[dl, "\mathsf{left}"'] \arrow[d, "\mathsf{val}"] \arrow[dr, "\mathsf{right}"] & \\ t & n & t \end{tikzcd}$$

and

$$\begin{tikzcd} & t^+ \arrow[dl] \arrow[dr, "\mathsf{incl}"] & \\ 1 \arrow[dr, "\mathsf{not}\circ\mathsf{true}"'] & & t \arrow[dl, "\mathsf{empty?}"] \\ & b & \end{tikzcd} \qquad (3)$$

be cones and that

$$\begin{tikzcd} 1 \arrow[r, "\mathsf{empty}"] \arrow[dr, "\mathsf{true}"'] & t \arrow[d, "\mathsf{empty}?"] \\ & b \end{tikzcd}$$

be a diagram.

In the cone (3), there should be an arrow from the vertex to the node b. It will appear in a model as either of the two (necessarily equal) composites. Since its value is forced, it is customary to omit it from the cone; however, there actually does have to be such an arrow there to complete the cone (and the sketch). In omitting it, we have conformed to the standard convention of showing explicitly only what it is necessary to show.

The existence of the first cone says that every nonempty tree can be represented uniquely as a triplet

$$(\mathsf{left}(T), \mathsf{val}(T), \mathsf{right}(T))$$

The fact that (3) is a cone requires that in a model M, $M(t^+)$ be exactly the subset of $M(t)$ of those elements which evaluate to false under $M(\mathsf{empty}?)$.

5.2 Comparison with logical language Logical language has no trouble expressing product theories, using multi-sorted signatures, although the notation is somewhat awkward. The equations can be expressed using derived operations. It can handle some limit theories, for example the torsion-free abelian groups, using universal Horn clauses, for example, $nx = 0 \Rightarrow x = 0$. But it would be quite awkward to express the theory of categories in logical language. It cannot be done using universal Horn clauses. The only way I know would be to begin with a relational language and express composition as a ternary relation, with universal Horn clauses to say it is a partial function and say what its domain is. However, relational theories are a much wider class of theories than limit theories, so this is an unwelcome step. Of course, you can express more theories this way, but there is less that you can say about each one.

5.3 Notational conventions We have been informally using some notational conventions by which most of the cones and diagrams in the sketch for semigroups above become uneccessary to specify explicitly. It would be possible to formalize them in such a way that they become part of the language, but I have refrained from doing so. The reason is that the main point of sketch is to replace the idea of a formal language by a category-based notion and reintroducing a formal language seems like a backward step. These notational conventions have the effect not of reducing the number of cones and diagrams, but rather of making them implicit in the notation, so they do not have to be spelled out in detail, but the underlying formalism remains the sketch.

Whenever a node in a sketch is called $s \times t$, it is assumed that there are nodes called s and t and a cone

$$
\begin{tikzcd}
 & s \times t \arrow[dl, "p_1"'] \arrow[dr, "p_2"] & \\
s & & t
\end{tikzcd}
$$

This can be generalized to more than two factors in an obvious way (even infinitely many). We may also write s^2, s^3, $\ldots$ for $s \times s$, $s \times s \times s$, $\ldots$, if it seems appropriate. A node called 1 is accompanied by a cone with vertex 1 and empty base. If $s \times t$ and $u \times v$ are nodes then an arrow $f \times g\colon s \times t \longrightarrow u \times v$ implies the existence of arrows $f\colon s \longrightarrow u$ and $g\colon t \longrightarrow v$ and the following diagram (this could obviously be broken into two, but the definition of commutative is sufficiently broad as to allow this diagram).

$$
\begin{tikzcd}
s \arrow[d, "f"'] & s \times t \arrow[l, "p_1"'] \arrow[r, "p_2"] \arrow[d, "f \times g"] & t \arrow[d, "g"] \\
u & u \times v \arrow[l, "p_1"] \arrow[r, "p_2"'] & v
\end{tikzcd}
$$

An arrow $\langle f, g\rangle\colon s \longrightarrow u \times v$ implies the existence of arrows $f\colon s \longrightarrow u$ and $g\colon s \longrightarrow v$ and a commutative diagram

$$
\begin{tikzcd}
 & s \arrow[dl, "p_1"'] \arrow[d, "\langle f, g\rangle"] \arrow[dr, "p_2"] & \\
u & u \times v \arrow[l, "f"] \arrow[r, "g"'] & v
\end{tikzcd}
$$

Finally, calling a node 1, implies that it is the vertex of a cone whose base is empty.

In a similar way, writing a node as $s \times_u t$ implies the existence of a cone of the form

$$
\begin{tikzcd}
 & s \times_u t \arrow[dl, "p"'] \arrow[d] \arrow[dr, "q"] & \\
s & u \arrow[l, "f"] \arrow[r, "g"'] & t
\end{tikzcd}
$$

in which f and g are already given arrows.

Later, we will add other notational conventions.

6 Sketches with cocones

The most expressive kind of sketch we will consider adds a set of cocones. So a **sketch with cocones** consists of $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{L}, \mathcal{C})$, where $(\mathcal{G}, \mathcal{D}, \mathcal{L})$ is a limit sketch and $\mathcal{C}$ is a class of cocones (the dual of cones) in $\mathcal{G}$. A model of this sketch is a model of the limit sketch that further takes all the cocones in $\mathcal{C}$ to colimit cocones. This concept is really too general and the main interest is in certain special cases, in which the class of cocones allowed is limited.

6.1 Regular sketches A ring R, not necessarily commutative, is called **regular (in the sense of von Neumann)** (since there is at least one other notion of regularity, at least for commutative rings), if for each $x \in R$ there is a $y \in R$ for which $xyx = x$. One way of making this into a categorical statement is as follows. Call such an element y a quasi-inverse of x. Should x happen to be invertible, then y is unique and is the inverse, but in general, there are many choices for y and no natural one. Given R, let $U \subseteq R \times R$ be the set of pairs of elements (x, y) for which $xyx = x$. We can define U by an equalizer

$$\begin{tikzcd} U \arrow[r] & R \times R \arrow[r, shift left, "p_1"] \arrow[r, shift right] & R \end{tikzcd}$$

in which p_1 is the first projection and the lower map, a derived operation, takes (x, y) to xyx. Then the composite $U \longrightarrow R \times R \xrightarrow{p_1} R$ has as image the set of elements that have at least one quasi-inverse. Thus R is regular iff that composite is surjective. Assuming we know what we mean by surjective, this would allow us to state regularity in any category that has the equalizer that defines U.

Now at this point, there is a problem, or rather a decision. There are at least three distinct categorial concepts that reduce to surjection in the case of sets. They are epimorphism, regular epimorphism, and split epimorphism (having a right inverse). It is not clear how to put the last one into the language of sketches (except by putting the right inverse into the sketch, which would then make the choice of right inverse part of the structure of a model, to be respected by homomorphisms, not what is wanted). The first choice can be sketched readily enough, but being an epimorphism is not a very good condition in a general category. On the other hand, regular epimorphisms are generally well-behaved and it is easy to describe them in the language of sketches. In fact, f is a regular epimorphism (in a category with suitable limits) iff there is a cone

$$\begin{tikzcd} & \cdot \arrow[dl, "d^0"'] \arrow[dr, "d^1"] & \\ \cdot \arrow[r, "f"'] & \cdot & \cdot \arrow[l, "f"] \end{tikzcd}$$

and a cocone

$$\begin{tikzcd}
\cdot \arrow[rr, shift left, "d^0"] \arrow[rr, shift right, "d^1"'] \arrow[dr] & & \cdot \arrow[dl, "f"] \\
& \cdot &
\end{tikzcd}$$

The first says that d^0 and d^1 are the kernet pair of f and the second that f is the coequalizer of d^0 and d^1. The reason this works is that in any category that has a kernel pair of f, the arrow f is a regular epimorphism iff it is a coequalizer of its kernel pair.

In general, regular sketches are used to translate the logical existential quantifier into sketch language.

6.2 Sum sketches A sketch $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{L}, \mathcal{C})$ is called a **sum sketch** if the cocones are discrete. Here is a sketch whose models in the category of sets is fields. We know that the category of fields cannot be sketched by a limit sketch since the category is not complete. On the other hand, there is a problem with the simple sketch I will now describe, which I will explain later.

6.3 Fields, take 1 The simplest way to describe a field is to say that it is a three sort theory, the first sort being that of the ring, the second being that of the multiplicative group of non-zero elements and third consisting of the 0 element alone. So we begin with a graph with nodes f, i, and z. We will need additional nodes such as $f \times f$, $f \times f \times f$ to express the fact that f is a ring. (Normally, we would also need f^0, to describe the additive and multiplicative units, but z can play that role). Similarly, we need $i \times i$ and $i \times i \times i$ and all the data needed to make i a commutative group. A map $\iota : i \longrightarrow f$ is needed and diagrams to say that that map preserves the multiplicative structure. The map $\zeta : z \longrightarrow f$ will already have been given to be the zero element of the additive group. Finally, we need a single discrete cocone:

$$\begin{tikzcd}
i \arrow[dr, "\iota"'] & & z \arrow[dl, "\zeta"] \\
& f &
\end{tikzcd}$$

to express the fact that every element is either 0 or invertible.

6.4 Fields, take 2 There are a couple of problems with this sketch. It is too set-theoretic. In intuitionistic logic, the various ways one might choose to express the axioms of a field are *not* equivalent. One can say, where $\mathrm{Inv}(x) = \exists y, xy = 1$

1. $\forall x, x = 0 \vee \mathrm{Inv}(x)$;
2. $\forall x, \neg x = 0 \Rightarrow \mathrm{Inv}(x)$;
3. $\forall x, \neg \mathrm{Inv}(x) \Rightarrow x = 0$;

and these are all different in intuitionisitic logic.

In a conversation, René Guitart, an Ehresmann student who has spent much of his career working on sketches, pointed out to me another objection. One of the nice features of limit sketches, in particular product sketches, is that the models in the category of topological spaces is just the category of topological models as traditionally understood. But if the simple sketch for fields is modeled in topological spaces, the fact that the field is the disjoint union of the invertible elements and 0 forces 0 to be isolated and then so is every element since topological groups are homogeneous. So a different approach is needed if we want to get non-discrete fields. It should be understood that we know what topological fields are; the point is to use this knowledge to gain insight and perhaps come up with the "right" definition of field. I no longer recall if the sketch I am about to give is the same as the one Guitart showed me, but it probably is and I give him full credit.

We begin with sorts f and i as above. We give f the operations and equations to be a commutative ring and u the operations and equations to be a commutative group. We then add sorts $\bar f$ and $\bar\imath$ along with z and build the sketch as above, replacing f be $\bar f$ and i by $\bar\imath$ so that $\bar f = \bar\imath + z$. To this we add operations $\bar f \longrightarrow f$, $\bar i \longrightarrow i$, and $i \longrightarrow f$ along with equations that make the first a ring homomorphism, the second a group homomorphism and the third a homomorphism of the group into the multiplicative monoid of the ring. Finally, we add cones and cocones that force the arrows $\bar f \longrightarrow f$ and $\bar\imath \longrightarrow i$ to be both monic and epic. This is done as follows for monics; epics are dual. An arrow $u\colon A \longrightarrow B$ is monic if and only if there is pullback square

$$\begin{tikzcd} A \arrow[r, "g"] \arrow[d, "g"'] & A \arrow[d, "u"] \\ A \arrow[r, "u"'] & B \end{tikzcd}$$

(and then g is forced to be an isomorphism). Thus we can put in a cone

$$\begin{tikzcd} & \bar f \arrow[dl] \arrow[d] \arrow[dr] & \\ \bar f \arrow[r, "\beta"'] & \bar f & f \arrow[l, "\beta"] \end{tikzcd}$$

Dually, we can force a map to be an epimorphism in a model.

Since epimorphisms in **Top** are surjective and monomorphisms are injective, a model of the resultant sketch will give a discrete field in bijective correspondence with a topological ring and that is the same as a topological field. Nonetheless the situation is not entirely satisfactory since in many categories bijections do not really correspond to anything interesting. Still this does give the right sketch in both sets and topological spaces.

6.5 Binary trees, again Here is a another sketch for binary trees that uses sums instead of the boolean test used above to distinguish the empty tree from the rest. Although this tree differs a lot from the sketch considered above (in this one, there can be only one empty tree; in the other one the set of empty trees in a model is arbitrary and can even be empty), they do have the same initial model (see below). Also it is not clear that an actual implementation would be that different. In any implementation, there would have to be a way of knowing if a tree were empty or not and this would presumably involve, implicitly or explicitly, some sort of boolean test or flag (which is just one way of doing the test quickly).

The sketch will have sorts $1,\ t,\ s,\ d$. Informally, t stands for tree, s for nonempty tree and d for datum. We have the following operations:

$\mathsf{empty}\colon 1 \longrightarrow t$

$\mathsf{incl}\colon s \longrightarrow t$

$\mathsf{val}\colon s \longrightarrow d$

$\mathsf{left}\colon s \longrightarrow t$

$\mathsf{right}\colon s \longrightarrow t$

The intended meaning of these operations is as follows.

Empty$\langle\rangle$ is the empty tree; incl is the inclusion of the set of nonempty trees in the set of trees; $\mathsf{val}(S)$ is the datum stored at the root of S; $\mathsf{left}(S)$ and $\mathsf{right}(S)$ are the right and left branches (possibly empty) of the nonempty tree S, respectively.

We require that

$$\begin{tikzcd} & s \arrow[dl, "\mathsf{left}"'] \arrow[d, "\mathsf{val}"] \arrow[dr, "\mathsf{right}"] & \\ t & d & t \end{tikzcd}$$

be a cone and that

$$\begin{tikzcd} 1 \arrow[dr, "\mathsf{empty}"'] & & s \arrow[dl, "\mathsf{incl}"] \\ & t & \end{tikzcd}$$

be a cocone.

There are no diagrams.

The cocone says that every tree is either empty or nonempty. This cocone could be alternatively expressed $t = s + \{\mathsf{empty}\}$. The cone says that every nonempty tree can be represented uniquely as a triplet $(\mathsf{left}(S), \mathsf{val}(S), \mathsf{right}(S))$ and that every such triplet corresponds to a tree. Note that this implies that left, val and right become coordinate projections in a model.

Using this, we can define subsidiary operations on trees. For example, we can define an operation of left attachment, $\mathsf{lat}\colon t \times s \longrightarrow s$ by letting

$$\mathsf{lat}(T, (\mathsf{left}(S), \mathsf{val}(S), \mathsf{right}(S))) = (T, \mathsf{val}(S), \mathsf{right}(S))$$

This can be done without elements: lat is defined in any model as the unique arrow making the following diagram commute (note that the horizontal arrows are isomorphisms):

$$\begin{tikzcd} M(t) \times M(s) \arrow[r, "M(t) \times \langle \mathsf{left}, \mathsf{val}, \mathsf{right} \rangle"] \arrow[d, "\mathsf{lat}"'] & M(t) \times M(t) \times M(d) \times M(t) \arrow[d, "\langle p_1, p_3, p_4 \rangle"] \\ M(s) \arrow[r, "\langle \mathsf{left}, \mathsf{val}, \mathsf{right} \rangle"'] & M(t) \times M(d) \times M(t) \end{tikzcd}$$

In a similar way, we can define right attachment as well as the insertion of a datum at the root node as operations definable in any tree. These operations are implicit in the sketch in the sense that they occur as arrows in the theory generated by the sketch and therefore are present in every model.

6.6 Proposition *Supposing there is an initial algebra for the data type, then the category of binary trees of that type has an initial algebra. If the data type has (up to isomorphism) a unique initial algebra, then so does the corresponding category of binary trees.*

Proof. We construct the initial algebra recursively according to the rules:

(i) The empty set is a tree;

(ii) If T_l and T_r are trees and D is element of the initial term algebra for the data type, then (T_l, D, T_r) is a nonempty tree;

(iii) Nothing else is a tree.

This is a model M_0 defined by letting $M_0(s)$ be the set of nonempty trees, $M_0(t) = M_0(s)+\{\emptyset\}$ and $M_0(d)$ be the initial model of the data type. Here '+' denotes disjoint union. It is clear how to define the operations of the sketch in such a way that this becomes a model of the sketch.

Now let M be any model with the property that $M(d)$ is a model for the data type. Then there is a unique morphism $f(d)\colon M_0(d) \longrightarrow M(d)$ that preserves all the operations in the data type. We also define $f(t)\{\emptyset\}$ to be the value of $M(\mathsf{empty})\colon 1 \longrightarrow M(t)$. Finally, we define

$$f(s)((T_l, D, T_r)) = (f(t)(T_l), f(d)(D), f(t)(T_r))$$

where $f(t)$ is defined recursively to agree with $f(s)$ on nonempty trees. It is immediately clear that this is a morphism of models and is unique. In particular, if the data type has, up to isomorphism, only one initial model then M_0 is also unique up to isomorphism. □

6.7 In Pascal textbooks a definition for a tree type typically looks like this:

```
type TreePtr = ^Tree;
     Tree = record LeftTree, RightTree :  TreePtr;
         Datum :  integer
     end;
```

Note that from the point of view of the preceding sketch, this actually defines nonempty trees. The empty tree is referred to by a null pointer. This takes advantage of the fact that in such languages defining a pointer to a type D actually defines a pointer to what is in effect a variant record (union structure) which is either of type D or of 'type' null.

7 Term models

A model M of a sketch with constants is called a **term model** if for every node a of the underlying graph, every element of $M(a)$ is reachable by beginning with constants and applying various operations (arrows of the sketch). The constants you begin with do not have to be of type a, but the final operation will, of course, have to be one that produces an element of type a. The significance of this condition from the computational point of view is that elements that cannot be produced in this way might as well not be there.

7.1 Example Let us consider models of the sketch described in 4.4. As mentioned there, one model of this sketch in **Set** is the natural numbers with the successor operation; the constant is 0. Other models are the integers and the integers modulo a fixed number k (in both cases, take the successor of x to be $x+1$). However, the natural numbers are the unique (up to unique isomorphism) **initial** model.

To see this, suppose M is any other model. Let us use the same letter M to denote $M(n)$ since there are no other nodes (common practice when the sketch has only one node). Also, let $t = M(\sigma): M \longrightarrow M$ and $m_0 = M(\zeta)$. We let $\mathbf{N}$, succ and 0 denote the values of these things in the natural numbers. To show that $\mathbf{N}$ is the initial model we must define a natural transformation $f: \mathbf{N} \longrightarrow M$ and show that it is the only one.

Define f as follows: let $f(0) = m_0$, as required if f is to be an arrow between linear sketches with constants. Then since f must commute with succ, we must have that $f(1) = t(m_0)$, $f(2) = t(t(m_0))$, and so on. This defines f inductively on the whole of $\mathbf{N}$. It is clearly unique and immediate to see that it is an arrow between models.

7.2 Example The set of all integers is a model but not a term model of the linear sketch of u-structures with one constant. For imagine you have a computer that can store integers, but the only operation that can be carried out on them is that of increment (successor). Suppose, further, that the only natural number whose existence you are certain of is 0. Then you can certainly produce, in addition, $1, 2, \ldots$, but no negative numbers. Therefore, they may as well not be there. You can get them by, for example, adding a decrement operation, but as it stands they are inaccessible. They are what J. Goguen and J. Meseguer have called ‘junk’.

7.3 Example The set $\mathbf{Z}_k$ of natural numbers $(\mathrm{mod}\, k)$ is a term model of the sketch of 4.4, but not an initial model. For example, there is no arrow from the natural numbers $(\mathrm{mod}\, k)$ to the natural numbers. In the first, the successor of $k-1$ is 0, while in the second it is nonzero. Thus no arrow could preserve successor at that point. What has happened here is that the model satisfies an additional equation $k = 0$ not required by the diagrams. This is an example of what Goguen and Meseguer call ‘confusion’.

7.4 Construction of initial term models Linear sketches with constants always have initial models. When the sketch is finite, an initial model can always be constructed recursively as a term model. (‘Finite’ means finite number of nodes and arrows.) We now give this construction.

Let $\mathcal{S} = (\mathcal{G}, \mathcal{D}, C)$ be a linear sketch with constants. We define a model $I: \mathcal{S} \longrightarrow$ **Set** recursively as the model constructed by the following requirements I–1 through I–3. The elements of $I(a)$ for a node a of $\mathcal{G}$ are congruence classes of **terms** of $\mathcal{G}$ (composable strings of arrows, including constants, of $\mathcal{G}$); $[x]$ denotes the congruence class of a term x by the congruence relation generated by the relation $\sim$ constructed recursively in the model. By 'congruence relation', we mean congruence on the free category generated by $\mathcal{G}$. In particular, if (g, f) and (g', f') are both composable pairs and $[f] = [f']$ and $[g] = [g']$, then $[g \circ f] = [g' \circ f']$.

I1. If a is a node of $\mathcal{G}$ and x is a constant of type a, then $[x] \in I(a)$.

I2. If $f\colon a \longrightarrow b$ is an arrow of $\mathcal{G}$ and $[x]$ is an element of $I(a)$, then $[fx] \in I(b)$ and $I(f)[x] = [fx]$. (Note that this constructs both an element and a value of the function $I(f)$ simultaneously.)

I3. If $(f_1, \ldots, f_m)$ and $(g_1, \ldots, g_k)$ are paths in a diagram in $\mathcal{D}$, both going from a node labeled a to a node labeled b, and $[x] \in I(a)$, then

$$(If_1 \circ If_2 \circ \ldots \circ If_m)[x] = (Ig_1 \circ Ig_2 \circ \ldots \circ Ig_k)[x]$$

in $I(b)$.

7.5 By 'the model constructed by' these requirements, we mean that

1. no element is in $I(a)$ except congruence classes of the terms constructed in I–1 and I–2, and
2. two terms are equivalent if and only if they are forced to be equivalent by the congruence relation generated by I–3.

Requirement 1 means that the models have no elements not nameable in the theory ('no junk') and 2 means that elements not provably the same are different ('no confusion'). It follows from requirement 2 that if $[x] = [y]$ in $I(a)$ and $f\colon a \longrightarrow b$ is an arrow of $\mathcal{G}$, then $[fx] = [fy]$ in $I(b)$.

Note that the models in 7.3 have terms giving the same element which are not forced to be equivalent by I–3.

7.6 Example Let us work out the initial term model of the sketch from 3.5 with one constant called x added. Since the sketch has only one node, the model has only one type. Thus in this case, there is only one set, call it S, and the arrows of the sketch lead to functions from S to S.

Then S has elements in accordance with the following rules:

Mod1. There is an element $[x] \in S$.

Mod2. If $[y] \in S$, then there are elements $[uy], [vy] \in S$.

Mod3. If $[y] = [z]$, then $[uy] = [uz]$ and $[vy] = [vz]$.

Mod4. For any $[y] \in S$, $[uvy] = [vuy] = [y]$.

It is clear that the set of all 'words' $[w_1 w_2 \ldots w_k x]$, where each w_i is either u or v, satisfies the first two rules above. In order to satisfy all four, we have to impose the equalities they force. In order to gain some insight into this, let us calculate some of the elements of S.

We observe that there must be elements

$$[x_0] = [x],\ [x_1] = [ux],\ [x_2] = [uux],\ \ldots\ ,\ [x_n] = \underbrace{[uu \cdots ux]}_{n \text{ copies}}$$

as well as elements we will denote

$$[x_{-1}] = [vx],\ [x_{-2}] = [vvx],\ \cdots\ ,\ [x_{-n}] = \underbrace{[vv \cdots vx]}_{n \text{ copies}}$$

We first explain why these elements exhaust S. We will not give a formal proof, but let us see which element is represented by an element chosen more or less at random, $[y] = [uvuvuuvux]$. Since $[vux] = [x]$, we have that $[y] = [uvuvuux] = [uvuvx_2]$. Since $[uvx_2] = [x_2]$, it follows that $[y] = [uvx_2]$ and then $[y] = [x_2]$, by another application of the same identity.

This kind of reasoning can be used to show that any application of u's and v's to $[x]$ gives the element $[x_k]$ where k is the number of u's less the number of v's.

In particular, $[ux_k] = [x_{k+1}]$ and $[vx_k] = [x_{k-1}]$ so the set $\{[x_k] \mid -\infty < k < \infty\}$ is carried into itself by both u and v. It contains $[x] = [x_0]$ and so must be all of S.

There remains the question of all the $[x_k]$ being distinct; that is whether or not there are any identities among the $[x_k]$. There is a standard way of resolving this question: if there is an equation among two combinations of arrows from the sketch, that equation must hold in every model. Thus if the equation fails in any one model, it cannot be a consequence of the identities in the sketch. In this case, there is an easy model, namely the set $\mathbf{Z}$ of all integers. In the set $\mathbf{Z}$, we let u act by addition of the number 1 and v act by subtracting 1. Then any combination of actions by u and v is just addition of k, the difference between the number of u's and v's (which may be negative).

The discussion above suggests how to construct a bijection between S and $\mathbf{Z}$ which is an isomorphism of models. We must choose an element to correspond to $[x]$. A plausible, but by no means necessary, choice is to correspond $[x]$ to 0. If we do that then we must correspond $[x_1] = [ux]$ to $[u0] = 1$, $[x_2] = [uux]$ to $[uu0] = 2$ and so on to correspond $[x_k]$ to k, for $k > 0$. For $k < 0$, the argument is similar, replacing u by v, to show that we correspond $[x_k]$ to k in that case as well.

The isomorphism just constructed takes each $[x_k]$ to the integer k, which implies that if $k \neq k'$, then $[x_k] \neq [x_{k'}]$. Thus S consists of precisely the distinct classes $[x_k]$, one for each integer $k \in \mathbf{Z}$.

7.7 Given the construction in 7.4 and any **Set** model M of the same sketch, the unique homomorphism $\alpha: I \longrightarrow M$ is constructed inductively as follows:

M1. If x is a constant of type a, then $\alpha a[x] = M(x)$.

M2. If $f: a \longrightarrow b$ in $\mathcal{G}$ and $[x] \in I(a)$, then $\alpha b([fx]) = M(f)([M(x)])$.

It is a straightforward exercise to show that this is well defined and is a homomorphism of models. It is clearly the only possible one.

> The construction in 7.4 can be seen as the least fixed point of an operator on models of the sketch (without the constants) in the category of sets and *partial* functions. To any such model M, the operator adjoins an element $f(x)$ to $M(b)$ for any arrow $f: a \longrightarrow b$ and any element $x \in M(a)$ for which $M(f)(x)$ is not defined. It forces $f(x)$ to be the same as some other element of $M(b)$ if the diagrams force that to happen (we leave the formal description of this to you). To get the model for a particular set of constants, you start with the model obtained by applying only I–1 (so that the sorts have only constants in them and all the arrows have empty functions as models). The least fixed point of this operator is the model in 7.4, up to isomorphism.

8 Initial term models for FP sketches

Just as in the case of a linear sketch with constants (see 7.4), a set-valued model of an FP sketch is called a **term model** if each element of the value at each node is forced to be there by the sketch. In the first instance, the node forced to be a terminal object is forced to have a unique element. But then by applying various operations, other elements are forced to exist. A term model has only those elements forced to exist in this way. This is spelled out precisely below in 8.4.

8.1 Definition A model of an FP sketch in an arbitrary category $\mathcal{C}$ is called an **initial model** or **initial algebra** if it has a unique homomorphism of models to each other model. It is thus an initial object in the category of models of the sketch in $\mathcal{C}$.

8.2 As with linear models with constants, an initial model of an FP sketch in the category of sets is necessarily a term model. For let M_0 be an initial model and suppose it is not a term model. Then the typed set of elements that are reachable beginning from the constants is certainly a model of the theory. It admits the constants and is closed, by definition, under the operations. Thus there is a term model $M_1 \subseteq M_0$. (The same argument, by the way, shows that every model includes a least submodel and that is a term model.)

Now we have an arrow (unique, actually) $f: M_0 \longrightarrow M_1$ by the definition of initial model. That arrow, composed with the inclusion, gives an arrow $M_0 \longrightarrow M_0$. But there is just one arrow from M_0 to itself, the identity (because M_0 is an initial model). Thus the composite must be the identity. But the image of f is included in M_1, so that $M_1 = M_0$, which means that M_0 is a term model, as claimed.

8.3 Another property any initial model must have is: if t and u are two terms definable starting with constants, then in an initial model M_0, $M_0(t) = M_0(u)$ if and only if $M(t) = M(u)$ in every model M. The nontrivial direction of that statement follows from the observation that if $M(t) \neq M(u)$ in some model, then necessarily $N(t) \neq N(u)$ in any model N for which there is a homomorphism $N \longrightarrow M$.

This property is described by saying that initial models have 'no confusion'. They are in fact characterized up to isomorphism by having no junk and no confusion. This terminology originated with J. Meseguer and J. Goguen.

8.4 Construction of initial models for finite FP sketches FP sketches always have initial models. We now revise the construction of 7.4 to construct initial models for finite FP sketches.

Let $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{L})$ be an FP sketch. In the construction of the initial model of a linear sketch with constants, we constructed terms as strings of arrows of $\mathcal{G}$. We will now allow tuples of arrows in these strings. The set $A_{\mathcal{S}}$ consisting of all arrows of $\mathcal{G}$, all tuples (of finite length) of such arrows and the cones C of $\mathcal{L}$ is called the **alphabet** of the sketch $\mathcal{S}$. The rules construct an initial model I recursively using these data. The rules apply to each cone C in $\mathcal{L}$ of the form

$$\begin{tikzcd} & & q \arrow[dll, "p_1"'] \arrow[d, "p_i"] \arrow[drr, "p_n"] & & \\ a_1 & \cdots & a_i & \cdots & a_n \end{tikzcd}$$

FP–1. If $f\colon a \longrightarrow b$ is an arrow of $\mathcal{G}$ and $[x]$ is an element of $I(a)$, then $[fx] \in I(b)$ and $I(f)[x] = [fx]$.

FP–2. If $(f_1, \ldots, f_m)$ and $(g_1, \ldots, g_k)$ are paths in a diagram in $\mathcal{D}$, both going from a node labeled a to a node labeled b, and $[x] \in I(a)$, then

$$(If_1 \circ If_2 \circ \ldots \circ If_m)[x] = (Ig_1 \circ Ig_2 \circ \ldots \circ Ig_k)[x]$$

in $I(b)$.

FP–3. If for $i = 1, \ldots, n$, $[x_i]$ is an element in $I(a_i)$, then $[C(x_1, \ldots, x_n)]$ is an element of $I(q)$. (Note that $C(x_1, \ldots, x_n)$ is a string consisting of the cone C followed by a tuple of arrows.) In particular, if $n = 0$, there is a single element $[C()]$ in the empty product.

FP–4. If for $i = 1, \ldots, n$, $[x_i]$ and $[y_i]$ are elements in $I(a_i)$ for which $[x_i] = [y_i]$, $i = 1, \ldots, n$, then
$$[C(x_1, \ldots, x_n)] = [C(y_1, \ldots, y_n)]$$

FP–5. For $i = 1, \ldots, n$, we require
$$[p_i C(x_1, \ldots, x_n)] = [x_i]$$

FP–6. For $x \in I(q)$, we require
$$[x] = [C(p_1 x, \ldots, p_n x)]$$

Thus FP–3 forces the vertex of the cone to contain an element representing each tuple of elements in the factors $a_1, \ldots, a_n$, and FP–5 forces the p_i to be the coordinate projections. Because it applies to empty cones, FP–3 subsumes I–1 of 7.4, which has no direct counterpart here. Observe that it follows from FP–1 and FP–5 that
$$I(p_i)[C(x_1, \ldots, x_n)] = [x_i]$$
for each i. FP–4 may be regarded as an extension of the definition of 'congruence relation' to cover the case of tuples.

8.5 Examples If M is the initial term model for an FP sketch $\mathcal{S}$, then for each sort g of the graph $\mathcal{G}$ of the sketch, $M(g)$ contains just those elements forced to be there by applying operations to constants. Indeed, up to isomorphism of models the elements *are* the formal applications of operations to constants, identifying those which the diagrams force to be the same.

Thus for the sketch in 4.4, in the intial model M, the value $M(1)$ must be a singleton set, and $M(\zeta)$ applied to that single element is an element of $M(n)$ which we may call 0. If we denote $M(n)$ by $\mathbf{N}$ and $M(\sigma)$ by $s\colon \mathbf{N} \longrightarrow \mathbf{N}$, then the elements of $\mathbf{N}$ are just 0, $s(0)$, $s(s(0))$, $s(s(s(0)))$, and so on. These can be identified as the natural numbers, starting at zero.

From this point of view, 4.4 is a simple data type description, and the initial model is then the set of possible values of that type.

Wells and Barr [1984] describe a class of FP sketches whose initial models are all context free grammars. Many other data types have been described using initial models using signatures and equations rather than FP sketches.

8.6 Free algebras Let $\mathcal{S}$ be a sketch with set S of nodes. An S-indexed set is a set X together with a typing function $\tau\colon X \longrightarrow S$. Our point of view is that the nodes of the sketch represent types and that X is a set of typed constants. If $\tau\colon X \longrightarrow S$ and $\tau'\colon X' \longrightarrow S$ are sets typed by the same sketch $\mathcal{S}$, then a function $f\colon X \longrightarrow X'$ is a **typed function** if $\tau = \tau' \circ f$. Sets typed by S and typed functions form the slice category $\mathbf{Set}/S$.

A particular example of a typed set is any model M of $\mathcal{S}$ in **Set** for which $M(c)$ and $M(d)$ have no elements in common for distinct nodes c and d. Any model is isomorphic to such a model, obtained by taking the disjoint union of the values of M at the different nodes of $\mathcal{S}$. A model is thus a family of sets, indexed by S, but we can as well think of it as a single set (the union) typed by S. In any case, the underlying (family of) set(s) of a model is an object of the slice category $\mathbf{Set}/S$.

Now given an S-indexed set X, let $\mathcal{S}_X$ be the sketch constructed by adding to the graph of $\mathcal{S}$ a set of arrows $x: 1 \longrightarrow s$ for each element $x \in X$ of type s. These are *in addition* to any constants of type S already given in the sketch. An initial model of $\mathcal{S}_X$, if one exists, is called the **free algebra** generated by the typed set X. We use the definite article because, although not unique, it is unique up to a unique isomorphism that preserves the set X for the same reason that initial algebras are always unique.

The following theorem gives the main existence result.

8.7 Theorem *Let $\mathcal{S}$ be an FP sketch. Then for any set X typed by the set of nodes of $\mathcal{S}$, there is a free algebra generated by X.*

Let $\mathcal{S}_X$ denote the sketch $\mathcal{S}$ augmented by X as a set of constants. This is still an FP sketch and as such has an initial model. It is not hard to see that this is a free model of $\mathcal{S}$ with X as generators. An accessible proof of this fact is in [Barr, 1986]. The free algebra on X is denoted $F(X)$.

8.8 Theorem *Let $\mathcal{S}$ be an FP sketch, X a typed set and M a model of $\mathcal{S}$ in sets. Then any typed function $f: X \longrightarrow M$ has a unique extension to an arrow between models $F(X) \longrightarrow M$.*

8.9 Example Let us see how our definition of free model allows us to discover that the free semigroup on a one element set is isomorphic to the semigroup $(\mathbf{N}^+, +)$.

Let the set be $\{a\}$. Thus we must construct the initial term model I of the sketch for semigroups with one arrow $1 \xrightarrow{a} s$ added; let us denote this sketch by $\mathcal{S}_a$. We write $c(x, y)$ as xy. It follows that $I(s)$ must have elements $[a]$, $[a][a]$, $[a][a][a]$, and so on. Let us call them simply a, a^2, a^3, $\ldots a^n, \ldots$. Now $(\mathbf{N}^+, +)$ is a model M of $\mathcal{S}_a$ with $M(s) = \mathbf{N}^+$, $M(c) = +$ and $M(a) = 0$. ($\mathbf{N}$ is also a model of $\mathcal{S}_a$ in other ways, but this is the way in which it is free on one generator.) Therefore by initiality there must be a unique homomorphism $h: I(s) \longrightarrow (\mathbf{N}^+, +)$ that takes a to 0. Then h necessarily takes a^n to n for each $n \in \mathbf{N}^+$. Since $a^m a^n = a^{m+n}$, there is also a homomorphism $k: (\mathbf{N}^+, +) \longrightarrow I(s)$ that takes n to a^n. It follows that h is an isomorphism with inverse k.

Note that it follows from this that there are no elements of $I(s)$ other than those of the form a^n. It is instructive to think about how you would prove this by a direct analysis of the sketch $\mathcal{S}_a$.

9 Term algebras for FD sketches

A complication arises in trying to extend the construction of initial term algebras to FD sketches. As we see from the examples of natural numbers and fields, an operation taking values in the vertex of a discrete cocone forces us to choose in which summand the result of any operation shall be. The choice, in general, leads to nonisomorphic term models which are nevertheless initial in a more general sense which we will make precise.

9.1 Dæmons How to make the choice? Clearly there is no systematic way. One way of dealing with the problem is to take *all* choices, or at least to explore all choices. In the example in of fields, not all choices are possible; once $2 \neq 0$, it followed that also $4 \neq 0$. (Recall that in that example, saying that something is zero is saying that it is in one of two summands.)

More generally, suppose we have an FD sketch and there is an operation $s\colon a \longrightarrow b$ and a cocone expressing $b = b_1 + b_2 + \ldots + b_n$. If we are building a model M of this sketch and we have an element $x \in M(a)$, then $M(s)(x) \in M(b)$, which means that we must have a unique i between 1 and n for which $M(s)(x) \in M(b_i)$. (For simplicity, this notation assumes that $M(b)$ is the actual union of the $M(b_i)$.)

Now it may happen that there is some equation that forces it to be in one rather than another summand, but in general there is no such indication. For example, the result of a push operation on a stack may or may not be an overflow, depending on the capacity of the machine or other considerations. Which one it is determines the particular term model we construct and it is these choices that determine which term model we will get.

Our solution is basically to try all possible sequences of choices; some such sequences will result in a model and others will abort. Thus as we explore all choices, some will eventually lead to a model; some will not. The theoretical tool we use to carry out this choice we call a **dæmon**. Just as a Maxwell Dæmon chooses, for each molecule of a gas, whether it goes into one chamber or another, our dæmon chooses, for each term of a model, which summand it goes into. The following description spells this out precisely.

9.2 Definition Let $\mathcal{S}$ be an FD sketch and suppose the maximum number of nodes in the base of any cocone is κ. A **dæmon** for $\mathcal{S}$ is a function d from the set of all strings in the alphabet $A_{\mathcal{S}}$ (see 8.4) of the underlying FP sketch (in other words, forget the cocones) to the initial segment $\{1\,..\,\kappa\}$ of the positive integers.

9.3 We will use a dæmon this way. We assume that the nodes in the base of each cocone of $\mathcal{S}$ are indexed by $1, 2, \ldots, k$ where $k \leq \kappa$ is the number of nodes in that cocone. In constructing an initial algebra, if a string w must be in a sort which is the vertex of a cocone (hence in the model it must be the disjoint union of no more than κ sorts), we will choose to put it in the $D(w)$th summand. If $D(w) > k$, the construction aborts. We will make this formal.

9.4 Construction of initial term models for FD sketches This construction includes the processes in 7.4 and 8.4; we repeat them here modified to include the effects of a dæmon D. The alphabet is the same as in 8.4.

If a node b is the vertex of a cocone with $k = k(b)$ summands, the summands will be systematically denoted $b^1, \ldots, b^k$ and the inclusion arrows $u^i : b^i \longrightarrow b$ for $i = 1, \ldots, k$. If b is not the vertex of a cocone, then we take $k(b) = 1$, $b^1 = b$ and $u^1 = \mathrm{id}_b$. We denote the congruence relation by $\sim$ and the congruence class containing the element x by $[x]$. Rules FD–1 through FD–3 refer to a cone C in $\mathcal{L}$ of the form:

$$\begin{tikzcd} & & q \arrow[dll, "p_1"'] \arrow[d, "p_i"] \arrow[drr, "p_n"] & & \\ a_1 & \cdots & a_i & \cdots & a_n \end{tikzcd}$$

FD–1. If $u^i : a^i \longrightarrow a$ is an inclusion in a cocone and $[x] \in I(a^i)$, then $[u^i x] \in I(a)$ and $I(u^i)[x] = [u^i x]$. (Thus we ignore the wishes of the dæmon in this case.)

FD–2. Suppose $f : a \longrightarrow b$ in $\mathcal{G}$ and $[x] \in I(a)$. Let $j = D(fx)$ (we ask the dæmon what to do). If $j > k(b)$, the construction aborts. Otherwise, we let $[fx] \in I(b^j)$ and $I(f)[x] = [u^j fx]$.

FD–3. For $i = 1, \ldots, n$, let $[x_i]$ be a term in $I(a_i)$. Let

$$j = D(C(x_1, \ldots, x_n))$$

If $j > k(q)$, the construction aborts. If not, put $[C(x_1, \ldots, x_n)]$ in $I(q^j)$.

FD–4. If for $i = 1, \ldots, n$, $[x_i]$ and $[y_i]$ are elements in $I(a_i)$ for which $[x_i] = [y_i]$, $i = 1, \ldots, n$, then

$$[C(x_1, \ldots, x_n)] = [C(y_1, \ldots, y_n)]$$

FD–5. For $i = 1, \ldots, n$, $I(p_i)([C(x_1, \ldots, x_n)]) = [x_i]$.

FD–6. For $x \in I(q)$,

$$[x] = [C(p_1 x, \ldots, p_n x)]$$

FD–7. If $\langle f_1, \ldots, f_m \rangle$ and $\langle g_1, \ldots, g_k \rangle$ are paths in a diagram in $\mathcal{D}$, both going from a node labeled a to a node labeled b, and $[x] \in I(a)$, then

$$(If_1 \circ If_2 \circ \ldots \circ If_m)[x] = (Ig_1 \circ Ig_2 \circ \ldots \circ Ig_k)[x]$$

in $I(b)$. If

$$D(f_1 f_2 \ldots f_m x) \neq D(g_1 \ldots g_k x)$$

(causing $(If_1 \circ If_2 \circ \ldots \circ If_m)[x]$ and $(Ig_1 \circ Ig_2 \circ \ldots \circ Ig_k)[x]$ to be in two different summands of $I(b)$), then the construction aborts.

This construction gives a term model if it does not abort. It is an initial model for only part of the category of models, however. In order to make this precise, we define a category to be **connected** if it is not possible to write it as a union of two proper subcategories with the property that there are no arrows between the objects in the one subcategory and the objects of the other subcategory. A **connected component** is a maximal connected set. It is readily seen that any category is a disjoint union of connected components. For example, in the category of fields, the connected components are determined by the characteristics. Two fields of the same characteristic may not have any arrows between them in either direction, but they are both the target of a unique arrow from the prime field of that characteristic. The next theorem says that that is typical behavior for the category of models of an FS theory.

9.5 Proposition *For each dæmon for which the construction in FD–1 to FD–6 does not abort, the construction is a recursive definition of a model I of $\mathcal{S}$. Each such model is the initial model of a connected component of the category of models of $\mathcal{S}$, and there is a dæmon giving the initial model for each connected component.*

We will not prove this theorem here. However, we will indicate how each model determines a dæmon which produces the initial model for its component. Let M be a model of an FD sketch $\mathcal{S}$. Every string w which determines an element of a sort $I(a)$ in a term model as constructed above corresponds to an element of $M(a)$. That element must be in a unique summand of a; if it is the ith summand, then define $D(w) = i$. On strings not used in the construction of the term models, define $D(w) = 1$, not that it matters.

Our definition of dæmon shows that one can attempt a construction of an initial model without already knowing models. In concrete cases, of course, it will often be possible to characterize which choices give initial models and which do not.

9.6 Confusion maybe, junk no Goguen's slogan, 'No junk, no confusion' is only half true of the initial models for FD theories. The 'No junk' half of the slogan expresses exactly what we mean when we say that every element is reachable. There are no extraneous elements. 'No confusion' means no relations except those forced by the equations in the theory. As we will show by example it may happen that some initial models have confusion and others not. Later we give an example of a sketch that has more than one unconfused initial model and one that has no unconfused initial (or noninitial) model.

If there is just one unconfused initial model, that one may be thought of as a 'generic' model. The others remain nonetheless interesting. In fact, it is likely that the generic model is the one that cannot be accurately modeled on a real machine.

9.7 Example A typical example of a sketch with many initial models is the sketch for natural numbers with overflow. The generic model is easily seen to be the one in 7.3 in which the overflow state is empty. The models with overflow in 9.7 are all initial algebras for some component of the category of models, but they have confusion, since nothing in the sketch implies that the successor of any element can be the same element. None of these models have junk.

The modular arithmetic models of 7.3 are not initial models; in fact they are all in the same component as the natural numbers since the remainder map (mod N) is a morphism of models. They also have no junk.

9.8 Example Here is a simple sketch with no generic model. It has two initial models, each satisfying an equation the other one does not. There are five nodes $a = b + c$, d and 1. There is one constant x of type d, and a single operation $s\colon d \longrightarrow a$. The initial models have one element – the constant – of type d. One of the initial models has an element of type b and the other an element of type c.

By modifying this example, we can get forced confusion. Add constants y and z of type b and c, respectively, and cones forcing b and c to be terminal. Now there are two initial models, one in which $s(x) = y$ and another in which $s(x) = z$. Since there is a model in which $s(x) \neq y$, there can be no equation that forces $s(x) = y$ and there is similarly no equation that forces $s(x) = z$. But one or the other equation must hold in any model.

9.9 Example It is well known and proved in abstract algebra texts that the initial fields are (a) the rational numbers and (b) the integers mod p for each prime p. (The word for initial model in these texts is 'prime field'.) A field is in the component of the integers mod p if and only if $1 + 1 + \ldots + 1$ (sum of p 1's) is zero. These fields have confusion. Otherwise the field is in the component of the rational numbers, which have no confusion (nor junk). Duval and Reynaud show how to implement simultaneous computation in the initial algebras for a finite discrete sketch for fields.

The real numbers and the complex numbers form fields with the usual operations. The irrational real numbers constitute junk.

References

M. Barr and C. Wells (1985), *Toposes, Triples and Theories.* Springer-Verlag, Berlin, Heidelberg, New York.

M. Barr (1986), Models of sketches. *Cahiers de Topologie et Géométrie Différentielle Catégorique*, **27**, 93–107.

M. Barr and C. Wells (1995), *Category Theory for Computing Science.* Prentice-Hall International, second ed.

A. Bastiani (1973), Sketched structures and completions, *Cahiers de Topologie et Géométrie Différentielle*, **14**, 158–160.

A. Bastiani and C. Ehresmann (1972), Categories of sketched structures. *Cahiers de Topologie et Géométrie Différentielle*, **13**, 104–213.

C. Ehresmann (1968), Esquisses et types des structures algébriques. *Bul. Inst. Polit. Iaşi*, **XIV**.

C. Ehresmann, (1969). Construction de structures libres, In *Category Theory, Homology Theory and their Applications II*, P. Hilton, editor, Lecture Notes in Mathematics **92**

F. E. J. Linton (1966), Some aspects of equational categories. *Proceedings of the Conference on Categorical Algebra at La Jolla.* Springer-Verlag, Berlin, Heidelberg, New York.

F. E. J. Linton (1969a), An outline of functorial semantics, *Lecture Notes in Mathematics* **80**, 7-52.

F. E. J. Linton (1969b), Applied functorial semantics, *Lecture Notes in Mathematics* **80**, 53-74.

M. Makkai and R. Paré (1990), *Accessible Categories: the Foundations of Categorical Model Theory*, *Contemporary Mathematics* **104**. American Mathematical Society.

C. Wells and M. Barr (1988), The formal description of data types using sketches. In *Mathematical Foundations of Programming Language Semantics*, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, Lecture Notes in Computer Science **298**