Adventures in Recreational Mathematics: Patterns of Patterns

Some definitions are better than others. Suppose I draw a list of operations $f_1, f_2, \ldots, f_n$ out of a hat and assign relations to them at random, like

$$\begin{aligned} &f_3(f_5(x,y),z,f_2(w)) = f_2(f_2(x)),\\ &f_4(f_2(x)) = f_1,\\ &f_3(x,x,y) = f_3(y,x,x), \end{aligned} \tag{1}$$

then I will have defined an algebraic object – call it a glarbl. However, glarbls will not be very nice objects: studying glarbls will not help you prove the Riemann hypothesis, calculate cohomologies, cook your breakfast, or anything else useful.

What is it that distinguishes the glarbl and its many cousins from normal, sane definitions like rings, groups, and tensor products? The answer to this question is obviously subjective, but ...

1 Compatibility

Nice things are nice to other nice things. Consider, for example, the definition of a derivation. Given a ring R and an R-module M, a derivation with coefficients in M is a map $D : R \to M$ satisfying

$$\begin{aligned} &D(0) = 0,\\ &D(1) = 0,\\ &D(x+y) = D(x) + D(y),\\ &D(-x) = -D(x),\\ &D(xy) = yD(x) + xD(y). \end{aligned} \tag{2}$$

Some rules can be dropped, but written in this way, the definition of a derivation has exactly one rule per each of the five ring operations $0, 1, +, -$, and $\cdot$.

Each rule takes D acting on the output of some n-ary ring operation f acting on free variables $x_1, \ldots, x_n$ and equates it with an expression in terms of those variables and $D(x_1), \ldots, D(x_n)$. Thus, the combined effect of these five rules which define a derivation is to give a canonical way to reduce D acting on any expression in terms of a set of variables $x_1, \ldots, x_n$ to an expression in terms of $x_1, \ldots, x_n, D(x_1), \ldots D(x_n)$.

Moreover, the reduction of expression is compatible with each of the ring axioms in such a way that equivalent expressions in the input of D get reduced to equivalent expressions in the output. For the associative law $(xy)z = x(yz)$, for example, we get

$$z(yD(x) + xD(y)) + xyD(z) = yzD(x) + x(zD(y) + yD(z)) \tag{3}$$

which holds true whether D is a derivation or not.

Expressions in $x_1, \ldots, x_n$ modulo the equivalences given by the ring axioms are just elements of the free ring $\mathbb{Z}[x_1, \ldots, x_n]$, meaning our definition of a derivation can be repackaged in the form of a map ϕ from the free ring $\mathbb{Z}[x_1, \ldots, x_n]$ to the free module generated by $D(x_1), \ldots, D(x_n)$.

Just about any definition that you might encounter in the wild satisfies this sort of compatability. One of the more common constructions is the tensor product of k-vector spaces: the tensor product is specified by the definition of a bilinear map and the definition of a bilinear map reduces to the map of free vector spaces

$$k[x_1, \ldots, x_n] \times k[y_1, \ldots, y_m] \to k[x_1 \otimes y_1, \ldots, x_n \otimes y_m] \tag{4}$$

given by applying the distributive law.

To formalize this notion of a definition compatible with existing operations, let us consider monads. For any purely algebraic object[1] there is the set of elements $F(S)$ of the free object generated by a set S, giving a functor $F : Set \to Set$. Each generator is also an expression, giving a natural transformation $\eta_S : S \to F(S)$. An expression in terms of other expression is also an expression itself, giving the 'multiplication' map $\mu_S : F(F(S)) \to F(S)$. The triple (F, η, μ) forms the structure of a monad. The actual objects equipped with the structure specified by F are the F-algebras. If you would like to further demand that all operations have only a finite number of arguments, this amounts to demanding that F commutes with filtered colimits.

More generally, for the allowance of structures that span multiple objects, you can consider monads over $\mathbf{Set}^{\times n}$ as well.

Definition 1.1. *Given a monad F on a category $\mathcal{C}$, let a compatible system over F be a monad G equipped with a natural transformation $\phi : G \circ F \to F \circ G$ such that it satisfies the obvious[2] compatibility condition with the monad structures on F and G.*

The compatibility conditions are such that $F \circ G$ naturally possess the structure of a monad. Let us call this monad the extension of F by G using ϕ.

[1] i.e. not counting fields

[2] That is, the conditions are not obvious but they're the only reasonable ones.

Example 1.2. *The ring monad is the extension of the Abelian group monad by the commutative monoid monad via the map which sends products of linear combinations of variables to their expansion into individual terms.*

Example 1.3 (Bilinear Maps)**.** *Let* $F : \mathbf{Set}^{\times 3} \to \mathbf{Set}^{\times 3}$ *be the functor*

$$(S, T, U) \mapsto (R^{\oplus S}, R^{\oplus T}, R^{\oplus U}), \tag{5}$$

where $R^{\oplus -}$ *is the free* R*-module functor.* F *is naturally equipped with a monadic structure.*

Then, we need to add a new operation with signature $\phi : M \times N \to L$*. If you have variables* $m_s \in M$ *for* $s \in S$*,* $n_t \in N$ *for* $t \in T$*, and* $l_u \in L$ *for* $u \in U$*, you can check that the only expressions that can be made with those symbols and the operation* ϕ *are the symbols themselves and expressions of the form* $\phi(m_s, n_t)$*. (Note that module operations like* $+$ *and* $\cdot$ *are not allowed here.)*

Thus, let $G : \mathbf{Set}^{\times 3} \to \mathbf{Set}^{\times 3}$ *be the functor sending*

$$(S, T, U) \mapsto (S, T, U \sqcup S \times T). \tag{6}$$

Equip G *with the unique reasonable monad structure* [3]*.*

Then, the definition of a bilinear map arises from the compatible system of G *and the natural transformation*

$$f_{(S,T,U)} : (R^{\oplus S}, R^{\oplus T}, R^{\oplus U} \sqcup R^{\oplus S} \times R^{\oplus T}) \to (R^{\oplus S}, R^{\oplus T}, R^{\oplus T \sqcup S \times T}) \tag{7}$$

which sends

$$\phi(r_1 m_1 + \cdots + r_s m_s, r'_1 n_1 + \cdots + r'_t n_t) \mapsto r_1 r'_1 \phi(m_1, n_1) + \cdots r_s r'_t \phi(m_s, n_t) \tag{8}$$

by applying the distributive law to expand out the expression. Compatibility of f *with the monadic structures on* F *and* G *arises from the fact that distributing out puts any such expression into a unique normal form.*

Note that the above is for a fixed R*. One re-run the above game with a generic ring where the monad* F *sends* (G, S, T, U) *to* $(\mathbb{Z}[G], \mathbb{Z}[G]^{\oplus S}, \mathbb{Z}[G]^{\oplus T}, \mathbb{Z}[G]^{\oplus U})$*.*

Proposition 1.4. *Ring structures over an abelian group, tensor products over a ring* R*, linear maps, homomorphisms of algebraic objects, derivations, p-derivations, etc. all arise from compatible systems.*

[3]These maps can be constructed explicitly by using the fact that the notation used is already correct. The unit sends n_u to itself. $G(G(A))$ consists of expressions made out of formal variables indexed by other expressions - denote the formal variable by expressions enclosed in brackets like $[\phi(m_s, n_t)]$. Then the multiplication map consists of dropping the brackets, sending $\phi([m_s], [n_t])$ and $[\phi(m_s, n_t)]$ to $\phi(m_s, n_t)$. This sort of principle provides monadic structure to any functor sending a set of variables to the set of formal expressions in those variables.

A classification of compatible extensions of common algebraic objects is probably interesting but too much work for me to do here.

2 Exactitude

Nice things generalize nicely to ∞-land. Monoids beget monoid objects, groups beget group objects, rings beget ring spaces, etc. Let us consider monads $F : \mathbf{Spc} \to \mathbf{Spc}$. When are these nice?

One condition we could demand is that F is 'finitary' in the sense that it commutes with filtered colimits. This effectively rules out operations with an infinite number of arguments. But what about F acting on the circle space S^1? This is an operation that acts on paths: it has a signature

$$\phi(x : A,\ h : x \sim x) : A. \tag{9}$$

There are interesting and natural monads which have operations with non-trivial path arguments[4] but let us instead consider what it would mean for there to be no such path operations.

Compatibility with filtered limits detect various finiteness conditions. The corresponding type of limit here is the hypercover.

Definition 2.1. *In the ∞-category* **Spc** *of homotopical spaces, let a hypercovering B of a space A be a simplicial object in* **Spc** *mapping into A*

$$\cdots B_1 \rightrightarrows B_0 \to A \tag{10}$$

such that the morphism

$$B[\Delta^n] \to B[\partial\Delta^n] \times_{\mathrm{Hom}(\partial\Delta^n, A)} A \tag{11}$$

is essentially surjective for all n.

The first two of these covering conditions are that $B_0 \to A$ is essentially surjective and that $B_1 \to B_0 \times_A B_0$ is essentially surjective.

The characteristic feature of the category of homotopical spaces is that all hypercoverings are also colimit diagrams. (In fact, I'm pretty sure that all diagrams of that shape are colimits if and only if they are hypercoverings.) We can now make a new definition in analogy to that of a finitary monad.

[4]Examples are E_n algebras and the n-truncation functor interpreted as a monad.

Definition 2.2. *An exacting monad is a monad on* **Spc** *(or more generally* $\mathbf{Spc}^{\times n}$*) which preserves hypercoverings.*

All homotopical spaces (and morphisms between them) can be covered by a diagram of sets[5]. Therefore, it should be the case that exacting monads are determined by their operation on the full subcategory of sets.

Conjecture 2.3. *A* **Set***-monad can be lifted to an exacting* **Spc***-monad if it preserves hypercovering diagrams in* **Set***.*

Theorem 2.4. *The monad of associative algebras is exacting.*

Proof. Given a set S, the free monoid $F(S)$ over S has an underlying set

$$\coprod_{n=0}^{\infty} S^{\times n}. \tag{12}$$

Then, on the n-th component, the hypercovering condition reduces to

$$B[\Delta^k]^{\times n} \to B[\partial\Delta^k]^{\times n} \times_{\mathrm{Hom}(\partial\Delta^n, A)^{\times n}} A^{\times n} = \left(B[\partial\Delta^k] \times_{\mathrm{Hom}(\partial\Delta^n, A)} A\right)^{\times n} \tag{13}$$

which is satisfied because the product of surjective morphisms is surjective. □

Conjecture 2.5. *The monads of Rings, Abelian groups, commutative monoids, modules, groups, etc. are all exacting. The glarbl monad is not exacting.*

The first nontrivial case is showing that the map $F(B \times_A B) \to F(B) \times_{F(A)} F(B)$ is surjective. In general, if two expressions in $F(B)$ map to the same expression in $F(A)$, they are related by a chain of multiple steps of substituting elements of B with equivalent elements and then rewriting the expression. We are now requiring that the expressions in $F(B)$ can be re-written in such a way that a single substitution of the variables relates the two expressions together.

Here is how this works in the case of abelian groups. Consider the free $\mathbb{Z}$-modules generated by $B = \{b_1, b_2, b_3, b_4\}$ and $A = a_1, a_2$ and let $f : B \to A$ be the function sending b_1 and b_2 to a_1 and sending b_3 and b_4 to a_2. In this case, the expressions $b_1 - b_2 + b_3$ and b_4 both get sent to the same element of $\mathbb{Z}^{\oplus A}$. A lift to $\mathbb{Z}^{\oplus B \times_A B}$ would be

$$[b_1, b_1] - [b_2, b_1] + [b_3, b_4]. \tag{14}$$

In genera, there seems to be enough room to add canceling terms in the nice monads so that the exacting requirements can be satisfied.

It would probably also be interesting to classify the low-complexity exacting monads. Have all of them already been discovered or can you find something new?

[5]See Kan complexes.