Burritos for the Hungry Mathematician

Ed Morehouse

April 1, 2015

Abstract

The advent of fast-casual Mexican-style dining establishments, such as Chipotle and Qdoba, has greatly improved the productivity of research mathematicians and theoretical computer scientists in recent years. Still, many experience confusion upon encountering burritos for the first time.

Numerous burrito tutorials (of varying quality) are to be found on the Internet. Some describe a burrito as the image of a crêpe under the action of the new-world functor. But such characterizations merely serve to reindex the confusion contravariantly. Others insist that the only way to really understand burritos is to eat many different kinds of burrito, until the common underlying concept becomes apparent.

It has been recently remarked by Yorgey [9] that a burrito can be regarded as an instance of a universally-understood concept, namely, that of monad. It is this characterization that we intend to explicate here. To wit, *a burrito is just a strong monad in the symmetric monoidal category of food, what's the problem?*

1 The Category of Food

The **category of food**, Füd, is a full subcategory of the category of stuff, Stf, whose objects are the stuff you can eat. A morphism of this category is known as a **recipe**. This category inherits the ambient *monoidal product*: if you can eat a carrot and you can eat a potato, then you can eat a carrot and a potato. The *monoidal unit* in the category of food is nothing, ε, which is to be found in many graduate students' refrigerators, typically, tensored with beer.

The monoidal category Füd is *symmetric*: if you can eat a carrot and a potato then you can eat a potato and a carrot. However, it fails to be *cartesian*. We lack the projections: if we have made the mistake of ordering a whiskey sour, there is generally no way to recover just the whiskey. Nor do we have a diagonal natural transformation, $\Delta : \mathrm{id} \longrightarrow - \otimes -$, giving us two whiskeys for the price of one – alas. For further details on the theory of categorical cookery, see [1].

2 The Tortilla Endofunctor

The category of food supports an endofunctor, $\mathrm{T} : \textsc{Füd} \longrightarrow \textsc{Füd}$, known as the **tortilla endofunctor**. Objects in the image of this functor are precisely those that are wrapped in a tortilla. The action of the functor T on morphisms is to take a recipe and yield a recipe from tortilla-wrapped ingredients to tortilla-wrapped results.

Note that the tortilla endofunctor is, in general, *not* monoidal. Given tortilla-wrapped rice and beans, it is usually not possible to recover tortilla-wrapped rice and tortilla-wrapped beans. Nor is it always possible to produce a tortilla-wrapped nothing out of nothing.

3 The Burrito Monad

It is universally understood that a **monad** on a category $\mathbb{C}$ is comprised of [6] an endofunctor $\mathrm{T} : \mathbb{C} \longrightarrow \mathbb{C}$ and two natural transformations, $\eta : \mathrm{id}(\mathbb{C}) \longrightarrow \mathrm{T}$ and $\mu : \mathrm{T}^2 \longrightarrow \mathrm{T}$, known respectively as the monad **unit** and **multiplication**, satisfying the monoid *unit* and *associative* laws:

$$(\eta \cdot \cdot \mathrm{T}) \cdot \mu \quad = \quad \mathrm{id}(\mathrm{T}) \quad = \quad (\mathrm{T} \cdot \cdot \eta) \cdot \mu \quad : \quad \mathrm{T} \longrightarrow \mathrm{T}$$

and

$$(\mu \cdot \cdot \mathrm{T}) \cdot \mu \quad = \quad (\mathrm{T} \cdot \cdot \mu) \cdot \mu \quad : \quad \mathrm{T}^3 \longrightarrow \mathrm{T}$$

In the case of the burrito monad, the functor in question is, of course, the tortilla endofunctor.

The burrito monad unit takes something you can eat and wraps it in a tortilla, yielding a **simple burrito**, which is both tortilla-wrapped and edible – no mean feat. This constitutes the familiar *insertion of generators* [5]. However, in the case of less-viscous generators, such as guacamole, some care may be required in the insertion, and it is often easier to insert the generator into the monad using a nozzle rather than by the obvious *folding map*.

The burrito monad multiplication takes a **compound burrito**, that is, something you can eat that is double-wrapped in tortillas, and merges the two tortillas into a single wrapping. This tends to happen spontaneously in the case of an overabundance of the previously-mentioned less-viscous generators, although a similar effect can be achieved mechanically by using some force and a *flattening function* (should we cite the crappy Gabor paper here?) [2].

Now we must check that the *monad laws* are satisfied:

monad right unit law: This law says that if you begin with something you can eat that's wrapped in a tortilla and you wrap the whole thing in another tortilla, then merge the tortillas, you get back what you started with, as the reader may deliciously verify.

monad left unit law: This law says that if you begin with something you can eat that's wrapped in a tortilla and you unwrap it, remove the contents

and wrap them in another tortilla, then wrap the result back up in the original tortilla and merge the two wrappings, the effect is again the identity function.

monad associative law: This law says that if you have a triple-tortilla-wrapped thing you can eat and you merge the inner two tortillas, then merge the result with the outer tortilla, the effect is the same as if you had merged the outer two tortillas first, and then the result with the inner one.

4 From Burritos, Strength

The burrito monad is in fact strong. A monad (T, η, μ) on a monoidal category $(\mathbb{C}, \otimes, \mathrm{I})$ is **strong** if there is a natural transformation,

$$\tau(\mathrm{A}, \mathrm{B}) \quad : \quad \mathrm{A} \otimes \mathrm{T}(\mathrm{B}) \longrightarrow \mathrm{T}(\mathrm{A} \otimes \mathrm{B})$$

satisfying certain relations [3]. In the case of a *strict* monoidal category, such as FÜD, these amount to the requirement that τ be a *morphism* of the monad structure. The corresponding laws are:

$$\mathrm{id}(\mathrm{A}) \otimes \eta(\mathrm{B}) \cdot \tau(\mathrm{A}, \mathrm{B}) = \eta(\mathrm{A} \otimes \mathrm{B}) : \mathrm{A} \otimes \mathrm{B} \longrightarrow \mathrm{T}(\mathrm{A} \otimes \mathrm{B})$$

and

$$\mathrm{id}(\mathrm{A}) \otimes \mu(\mathrm{B}) \cdot \tau(\mathrm{A},\mathrm{B}) = \tau(\mathrm{A},\mathrm{T}(\mathrm{B})) \cdot \mathrm{T}(\tau(\mathrm{A},\mathrm{B})) \cdot \mu(\mathrm{A} \otimes \mathrm{B}) : \mathrm{A} \otimes \mathrm{T}^2(\mathrm{B}) \longrightarrow \mathrm{T}(\mathrm{A} \otimes \mathrm{B})$$

In the case of the burrito monad, the strength, τ, is the "wrapping in" transformation, which takes a "side" object A and a burrito T(B) and yields a burrito in which the side has been wrapped into the burrito. This is most clearly illustrated by an example. Suppose that you begin with a side of guacamole and a bean burrito. By "wrapping in" the side, you may obtain a guacamole and bean burrito – which is clearly better. In this case, the monad strength laws require the following:

strength unit law: If you first insert some beans into a simple burrito and next wrap-in a side of guac, the effect is the same as if you had inserted the guac and beans into a simple burrito together.

strength multiplication law: If you have a compound bean burrito that you first merge into a simple one and you then wrap-in a side of guac, the effect is the same as if you had wrapped the guac into the outer, bean burrito, burrito, then wrapped the (now interstitial) guac into the inner, bean, burrito, and then finally merged the compound guac and bean burrito

into a simple one. This last equivalence is perhaps best understood as:

	guac and ((bean burrito) burrito)		guac and ((bean burrito) burrito)
⇒	[merge compound burrito]	⇒	[wrap-in guac to outer burrito]
	guac and (bean burrito)		(guac and (bean burrito)) burrito
⇒	[wrap-in guac]	= ⇒	[wrap-in guac to inner burrito]
	(guac and bean) burrito		((guac and bean) burrito) burrito
		⇒	[merge compound burrito]
			(guac and bean) burrito

5 Functional Burritos

Those coming to burritos from a functional programming background may prefer to understand them by way of the "return and bind" formulation of monads [8], popularized by the *typeclass* mechanism of languages such as Haskell.

From this perspective, the burrito monad's `return` is just the same as its unit η. Flipping argument order, we see that the `bind` operation, $(\mathrm{A} \to \mathrm{TB}) \to \mathrm{TA} \to \mathrm{TB}$, transforms a burrito recipe into one that expects to receive its ingredients in burrito form. The monad laws in this formulation are satisfied just in case one takes `flip bind` f to be $\mathrm{T}(f) \cdot \mu(\mathrm{B})$.

The T-image of the morphism f performs f while wrapped in a tortilla. Depending on the nature of the morphism, this may raise size issues. In the preceding discussion, we have assumed all burritos to be at least locally small.

6 Conclusion and Related Work

As you can see, burritos are not so hard to understand, once viewed from the proper perspective – namely, that of category theory. We hope that this little tutorial will encourage more researchers to incorporate burrito-theoretic results and methods into their own work.

Burrito-like sandwiches have emerged several times independently, but were not put on a solid theoretical footing until their connection to monads became apparent in the early 1970s [3, 7]. More recently, a connection to the *Hegelian taco*, a display of the 3-dimensional eight-element graphic monoid with five left ideals, has been worked out by Lawvere [4]. It is conjectured that weak equivalences exist to other wrap-like structures, but research in this area is presently ongoing.

References

[1] Bob Coecke. "Quantum Picturalism". In: *Contemporary Physics* (2009). URL: `http://arxiv.org/abs/0908.1787`.

[2] Cat Ferguson. *Overly honest references: "Should we cite the crappy Gabor paper here?"* URL: http://retractionwatch.com/2014/11/11/overly-honest-references-should-we-cite-the-crappy-gabor-paper-here/.

[3] Anders Kock. "Strong Functors and Monoidal Monads". In: *Archiv der Mathematik* 23 (1972), pp. 113–120.

[4] F. William Lawvere. "Display of Graphics and their Applications, as Exemplified by 2-Categories and the Hegelian Taco". In: *Proceedings of the First International Conference on Algebraic Methodology and Software Technology*. 1989.

[5] Saunders Mac Lane. *Categories for the Working Mathematician*. second edition. Graduate Texts in Mathematics. Springer, 1998.

[6] Mark Molloy. *Grammar crusader spends years removing repeated error 47,000 times on Wikipedia*. URL: http://www.telegraph.co.uk/men/the-filter/11392756/Grammar-crusader-spends-years-removing-repeated-error-47000-times-on-Wikipedia.html.

[7] Ross Street. "The Formal Theory of Monads". In: *Journal of Pure and Applied Algebra* 2 (1972), pp. 149–168.

[8] Philip Wadler. "Monads for Functional Programming". In: *Advanced Functional Programming*. Lecture Notes in Computer Science 925. Springer-Verlag, 1995.

[9] Brent Yorgey. *Abstraction, intuition, and the "monad tutorial fallacy"*. URL: https://byorgey.wordpress.com/2009/01/12/abstraction-intuition-and-the-monad-tutorial-fallacy/.