CATEGORICAL ASPECTS OF SEMANTICS

Lecture Notes

Jürgen Koslowski

Theoretical Computer Science

Technische Universität Braunschweig

2018-06-06

Contents

0 Alternating automata revisited

0.0 Ordinary automata and their finiteness conditions

Finite automata for recognizing languages over some alphabet A may be viewed as labeled transition systems (LTSs) on some set Q of states, equipped with initial and final states. The LTS's transition function δ may be formulated in various ways, but from a categorical perspective a very sensible choice is to map the alphabet A into a suitable monoid:

$$\begin{array}{ll} A \xrightarrow{\delta} \langle Q, Q\rangle \boldsymbol{set} & \text{in the total deterministic case} \\ A \xrightarrow{\delta} \langle Q, Q+1\rangle \boldsymbol{set} = \langle Q, Q\rangle \boldsymbol{prt} & \text{in the deterministic case} \\ A \xrightarrow{\delta} \langle Q, QP\rangle \boldsymbol{set} = \langle Q, Q\rangle \boldsymbol{rel} & \text{in the nondeterministic case} \end{array}$$

Here we observe that both $\boldsymbol{prt}$, the category of sets and partial functions, as well as $\boldsymbol{rel}$, the category of sets and binary relations (ignoring, for the moment, the 2-cells in both cases) are isomorphic to the Kleisli-category $\boldsymbol{set}_{\mathbb{E}}$ for the exception moand $\mathbb{E} = \langle - + 1, \eta_{\mathbb{E}}, \mu_{\mathbb{E}}\rangle$, respectively, $\boldsymbol{set}_{\mathbb{P}}$ for the power set monad $\mathbb{P} = \langle P, \{-\}, \bigcup\rangle$. This clarifies the monoid structure of the codomain of δ, we were aiming for, and which is perhaps not immediate when looking at the hom-sets $\langle Q, Q+1\rangle \boldsymbol{set}$ and $\langle Q, QP\rangle \boldsymbol{set}$.

But careful, these formulations do not express the finiteness requirements usually imposed in automata theory, namely that there are only finitely many labeled transitions, *i.e.*, the disjoint union of the sets $a\delta \subseteq Q \times Q$, $a \in A$, has to be finite.

In case of total deterministic automata this means that alphabet and state-set are finite, while in the other two cases only the set of letters to which non-empy relations are assigned has to be finite. Moreover, the relations that are assigned have to be finite. In case of non-deterministic automata that suggests replacing the full power-set functor P with F, which maps sets to their sets of finite subsets.

$$A \xrightarrow{\delta} \langle Q, QF\rangle \boldsymbol{set} = \langle Q, Q\rangle \boldsymbol{set}_{\mathbb{F}} \quad \text{in the nondeterministic case}$$

This gives rise to a sub-monad $\mathbb{F}$ of $\mathbb{P}$. Notice, however, that while its Kleisli-category $\boldsymbol{set}_{\mathbb{F}}$ is no longer self-dual, *i.e.*, not closed under the reversal $(-)^{\mathrm{op}}$ of relations, the constraint of only allowing finitely many transitions altogether still allows the reversal of automata.

It would be desirable if the initial and final states can also be specified in the same setting that allowed us to idemtify the monoid structure. Indeed, for initial states one can use a total function $1 \xrightarrow{I} Q$ that specifies a single initial state, a partial function $1 \xrightarrow{F} Q$ that specifies at most one initial state, and a relation $1 \xrightarrow{F} Q$ that specifies a set of initial states. In the latter two cases a dual construction can be used to specify final states: a partial function $Q \xrightarrow{F} 1$ and a relation $Q \xrightarrow{F} 1$ both specify a set of final states. Hence the recognized words can be characterized as those where the composite of I with the transition relations and F results in the non-empty relation on 1. In the total deterministic case this is not possible; to specify a subset $I \subseteq Q$ we need to employ 2 as the codomain of the characteristic function $I\chi$.

This inherent asymmetry may have contributed to the development of coalgebra, which combines an exponential transpose $Q \xrightarrow{d} \langle A, Q\rangle\boldsymbol{set}$ of $A \xrightarrow{\delta} \langle Q, Q\rangle\boldsymbol{set}$ with $I\chi$ to obtain a coalgebra $Q \xrightarrow{D} \langle A, Q\rangle\boldsymbol{set} \times 2$.

0.1 Alternating automata

The intuition for alternating automata is to complement the aspect of choice provided by functions $Q \longrightarrow QF$ with an aspect of parallelism, or maybe superposition. This can be achieved by replacing QF by the set QB of *positive Boolean combinations* of elements of Q.

$$A \xrightarrow{\delta} \langle Q, QB\rangle\boldsymbol{set} \quad \text{in the alternating case}$$

But what kind of monoid structure can we impose on $\langle Q, QB\rangle\boldsymbol{set}$? Recall that every Boolean formula has a conjunctive as well as a disjunctive normal form, both can be written as finite sets of finite sets of literals. In the positive case only atoms occur. Interpreting a finite set of finite sets of states as a positive DNF, the outer power-set constructor is responsible for choice, and the inner one for superposition.

This raises the question, whether the monad $\mathbb{F}$ can be composed with itself to yield a monad $\mathbb{B}$. In that case we can identify $\langle Q, QB\rangle\boldsymbol{set}$ with $\langle Q, Q\rangle\boldsymbol{set}_{\mathbb{B}}$, which then carries the required monoid structure. As we have seen before, this would require a distributive law linking $\mathbb{F}$ with itself.

0.2 The monad $\mathbb{B}$

The distributive law in question does in fact express the fact that in Boolean algebra conjunction distributes over disjunktion, and vice versa. To transform a Boolean formula from CNF into DNF, form the disjunction of those co-clauses that result from picking one literal from every clause of the CNF. More formally, form the cartesian product of all co-clauses, and take the images of the resulting choice-functions as new co-clauses. The same transformations works in the other direction, since conjunction distributes over disjunction as well.

The unit of this monad maps $q \in Q$ to singleton $\{\{q\}\}$, while the multiplication is somewhat messy to describe: interpret an element $\mathcal{A} \in QFFFF$ as the disjunction of a set of formulae in CNF, over conjunctions of literals rather than literals. Turning the central CNF into a DNF results in a disjunction of set of disjunctions of conjunctions of further conjunctions, and hence another DNF.

Since the EM-algebras of the monad $\mathbb{F}$ are either $\sqcap$-or $\sqcup$-semilattices, depending on the chosen order on subsets, the EM-algeras of $\mathbb{B}$ combine both operations in such a way that they distribute over one another, hence turn out to be distributive lattices (with $\top$ and $\bot$, since the empty set is allowed at both levels). The Kleisli-category $\boldsymbol{set}_{\mathbb{B}}$ is known to be equivalent to the subcategory of free distributive lattices. In the Kleisli category we just keep track of the generators, which form an anti-chain about which the free distrubutive lattice is symmetric.

Note that $\emptyset$ generates the 2-element chain with $\bot < \top$ in this fashion, while 1 generates a 3-element chain $\bot < * < \top$. To continue the analogy with ordinary automata above, one would expect $\boldsymbol{set}_{\mathbb{B}}$-morphisms $1 \xrightarrow{I} Q$ amd $Q \xrightarrow{F} 1$ to specify a set of initial superpositions, and a setof final states. Indeed, the corresponding function $1 \xrightarrow{I} QB$ picks out an element of the free distributive lattice over Q, while for $Q \xrightarrow{F} 1B$ we can consider the pre-image of $\top$ as the set of final states.

An infinitray version of essentially the same distributive law is available for composing the power-set monad $\mathbb{P}$ with itself; in that case one obtains completely distributive lattices as EM-algebras.

0.3 Moving to the ordered setting

The distributive laws outlined above make sense in the category $\boldsymbol{ord}$ of antisymetrically ordered sets, which subsumes the category $\boldsymbol{set}$, where equality serves as a discrete order. Recall that $\boldsymbol{ord}$ is a full subcategory of $\boldsymbol{pre}$, the category of $\mathbf{2}$-enriched categories, which means reflexive and transitive relations, $\mathbf{2}$ being the 2-element chain with $0 < 1$. In particular, any order-relation $\sqsubseteq$ on a set Q can be viewed as an order-preserving function $\boldsymbol{Q}^{\mathrm{op}} \times \boldsymbol{Q} \xrightarrow{\sqsubseteq} \mathbf{2}$, the hom-functor for $\boldsymbol{Q}$, which among *order ideals* plays the role of the identity on $\boldsymbol{Q}$.

In this setting the power-set functor splits into two variants by fixing either of the two slots of the hom-functor with $\mathbf{2}$: the down-set functor D is just $\langle (-)^{\mathrm{op}}, \mathbf{2} \rangle \boldsymbol{ord}$, while the up-set functor U is $(\langle -, \mathbf{2} \rangle \boldsymbol{ord})^{\mathrm{op}}$.

The ordering has been chosen to be able to embed $\boldsymbol{Q}$ into these ordered sets (Yoneda embedding); just recall how the principal up- and down-sets reflect $x \sqsubseteq y$. This implies $x{\downarrow} \subseteq y{\downarrow}$ and $x{\uparrow} \supseteq y{\uparrow}$.

$$\boldsymbol{Q} \xrightarrow{\downarrow} \boldsymbol{Q}D = \langle \boldsymbol{Q}^{\mathrm{op}}, \mathbf{2} \rangle \boldsymbol{ord} \quad \text{resp.} \quad \boldsymbol{Q} \xrightarrow{\uparrow} \boldsymbol{Q}U = (\langle \boldsymbol{Q}, \mathbf{2} \rangle \boldsymbol{ord})^{\mathrm{op}}$$

The units of the corresponding monads map $q \in Q$ to its *principal down-* respectively *up-set*, while the multiplications maps an up-set of down-sets to its union, and similarly a down-set of up-sets. EM-algebras of $\mathbb{D}$ and $\mathbb{U}$ are $\bigsqcup$-lattices, resp., $\bigsqcap$-lattices

These functors obey generalizations of the distributive laws above, and may be composed with each other in both ways, yielding the same new monad. Its EM-algebras are known as *constructively completely distributive lattices*, or *CCD-lattices* for short. [Literature: Fawcett, Wood, Rosebrugh, Marmolejo...]

Originally, CCD-lattices were defined in analaogy to continuous lattices by means of a so-called "way-below" relation. It turns out that a poset $\boldsymbol{Q}$ is a complete lattice, iff the Yoneda embedding $\boldsymbol{Q} \xrightarrow{\downarrow} \langle \boldsymbol{Q}^{\mathrm{op}}, \mathbf{2} \rangle \boldsymbol{ord}$ has a left adjoint $\bigvee$, which turns out to be a right-inverse as well (*cf.* the notion of EM-algebra). A complete lattice is CCD, if $\bigvee$ has a further left adjoint $\boldsymbol{Q} \xrightarrow{\downarrow} \boldsymbol{Q}D$ that maps $q \in Q$ to the smallest down-set with supremum q. Its elements are said to be "way-below" q. The way-below-function turns out to be automatically left-inverse to $\bigvee$.

If one replaces down-sets by directed down-sets, sometimes called ideals, one obtains a similar characterization of continuous lattices: every principal down-set is of course directed, and the existence of a left adjoint (and automatically right-inverse) to the Yoneda embedding into the ideal-poset is equivalent to directed completeness: every directed set has a supremum. And the existence of a further left-adjoint left-inverse to $\bigvee$ characterizes continuous lattices. (As far as I know there is no distributive law linking (up-) directed down-sets with down-directed up-sets.)

1 Categorical Aspects of the typed λ-calculus

1.0 A problem with the untyped λ-calculus

The untyped λ-calculus (Alonso Church) can be seen as an attempt by logicians to treat functions as first-class citicens.

The idea seems to be to turn "rules" into functions, which results in an "intensional" theory of functions, as opposed to the "extensional" one that results from interpreting functions as special relations, hence ultimately as sets (often called "graphs"). In the latter case two functions are equal, if the corresponding graphs are equal as sets. The situation is more subtle in the λ-calculus. An additional complication arises from the fact that the term "intensional" is used differerntly by mathematicians and by philosophers, see, *e.g.*, Stanford Encyclopedia of Philosophy.

In the untyped λ-calculus one can define *e.g.*, Boolean truth-values and junctors including n, Church-numerals and pairs with projections. In fact, the untyped λ-Calculus is Turing-complete.

Unfortunately, in 1935 Church and Rossner discovered a paradox: Defining

$$r = \lambda x.(\neg(xx))$$

seems innocent enough, but when applied to itself results in

$$rr = \lambda x.(\neg(xx)r = \neg(rr)$$

For logicians this was a major problem and prompted Church to define the simply typed λ-calculus that avoided self-reference (this had been successful in solving the Russel-Whitehead paradox in set-theory. [consider also Chapter 6 of "The Lambda Calculus", in the Stanford Encyclopedia of Philosophy].

From a computer scientists point of view the apparent paradox is just a non-terminating computation, and as such not life-threatening.

1.1 The simply typed λ-calculus

Besides avoiding self-reference, the simply typed λ-calculus had another advantage over its untyped cousin: it was strongly normalizing, so every expression is guaranteed to terminate.

This, of course, comes at a price: the loss of Turing-completeness. Otherwise one could *e.g.* disprove Goldbach's conjecture that every even integer is the sum of two primes: just write a program that looks for even intergers *not* having this property and halting then. Without even running this program, it would be guaranteed to halt.

The simply typed λ-calculus $\Lambda^{\to}$ is a 2-sorted theory based on *types* and *terms* or *expressions*. Starting from a set B of *base types*, the BNF for types is given by

$$\tau ::= \tau \to \tau \,|\, T \in B$$

Often just a single gound type o is used, *i.e.*, $B = \{o\}$, while in other cases one has ground types $\boldsymbol{Bool}$ and $\boldsymbol{Nat}$, etc.. Strangely, there is no 0-ary application of $\to$.

To reduce parentheses, the operator $\to$ is supposed to associate to the right, *i.e.*, $\sigma \to \tau \to \chi$ means $\sigma \to (\tau \to \chi)$.

Terms come in four varieties: variables, abstractions, applications and constants from a given set C:

$$e ::= x \,|\, \lambda x : \tau.e \,|\, ee \,|\, c \in C$$

Again, to reduce parenthesis, application (written as concatenation) associates to the left, *i.e.*, xyz means $(xy)z$.

Here $x : \tau$ means "x is of type τ". In fact, every expression is assigned a unique type, either in advance for the constants (*e.g.*, 0 of type $\boldsymbol{Nat}$ or $\boldsymbol{s}$ of type $\boldsymbol{Nat} \to \boldsymbol{Nat}$, or recursively according to the following *typing rules*. Typing assignments are typically written in a notation resembling Gentzen's sequent calculus, where capital greek letters Γ, Δ ... represents the environment of known type assignments up to this point.

$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma} \qquad \frac{c \in C \text{ of type } T}{\Gamma \vdash c : T}$$

$$\frac{\Gamma, x : \sigma \vdash e : \tau}{\Gamma \vdash (\lambda x : \sigma.e) : (\sigma \to \tau)} \qquad \frac{\Gamma \vdash e_0 : (\sigma \to \tau) \;\; \Gamma \vdash e_1 : \sigma}{\Gamma \vdash e_0 e_1 : \tau}$$

Terms that can be assigned types in the empty context are called *combinators*.

▷ For every type τ the *identity combinator* $\boldsymbol{I}_\tau$ is given by the term

$$(\lambda x : \tau.x)(\tau \to \tau)$$

▷ For any types σ, τ the combinator $\boldsymbol{K}_{\sigma,\tau}$ is given by the term

$$(\lambda x : \sigma.\lambda y : \tau.x) : (\sigma \to \tau)$$

▷ For any types σ, τ and χ, the combinator $\boldsymbol{S}_{\sigma,\tau,\chi}$ is given by the term

$$\lambda x : (\sigma \to \tau \to \chi).\lambda y : (\sigma \to \tau).\lambda z : \sigma.xz(yz) : ((\sigma \to \tau \to \chi) \to (\sigma \to \tau) \to \sigma \to \chi)$$

Notice that similarity of the typing of $\boldsymbol{K}_{\sigma,\tau}$ and of $\boldsymbol{S}_{\sigma,\tau,\chi}$ with the axioms (Ax1) and (Ax2) of the Hilbert-style deductive system $\mathcal{F}_0$! In fact, I_τ is derivable from these, since the corresponding proof in $\mathcal{F}_0$ does not use (Ax3).

Finally, there are rewriting-rules:

Interpeting the types as objects and the equivalence classes of terms as morphisms, one should obtain a category $\mathcal{L}$ with extra structure in form of a functor $\mathcal{L}^{\mathrm{op}} \times \mathcal{L} \xrightarrow{\triangleright} \mathcal{L}$ subject to certain axioms (see below). There is an obvious notational problem as the type-building operator $\to$ may clash with the categorical arrow $\longrightarrow$ for arrows (although in TEX@ we can use different types of arrows, on the blackbord or in handwritten notes this can be a serious problem.)

1.2 The simply typed λ-calculus with products

Often the simply typed λ-calculus is enhanced by additional type-bilding operators, most importantly (cartesian) products and the corresponding projections.

While in the untyped λ-calculus pairs and projections can be derived, this does not seem to work in the simply typed λ-calculus, see here for an interesting discussion and a fallacy concerning coporoducts, *i.e.*, disjoint unions.

Adding products also forces consideration of the empty product, called *unit type*, which was somehow missing above.

1.2.1 The unit type

$$\tau ::= \tau \to \tau \,|\, Unit \,|\, T \in B$$

$$e ::= x \,|\, \lambda x \,:\, \tau.e \,|\, ee \,|\, () \,|\, c \in C$$

$$\frac{}{\Gamma \vdash () : Unit} \qquad \frac{\Gamma \vdash e_0 : \sigma \quad \Gamma \vdash e_1 : Unit}{\Gamma \vdash e_0 e_1 : \sigma}$$

1.2.2 Pairs, the product type

$$\tau ::= \tau \to \tau \,|\, Unit \,|\, \sigma \times \tau \,|\, T \in B$$

$$e ::= x \,|\, \lambda x \,:\, \tau.e \,|\, ee \,|\, (e,e) \,|\, \mathrm{fst}\, e \,|\, \mathrm{snd}\, e \,|\, () \,|\, c \in C$$

$$\frac{\Gamma \vdash e_0 : \sigma \quad \Gamma \vdash e_1 : \tau}{\Gamma \vdash (e_0, e_1) : \sigma \times \tau} \qquad \frac{\Gamma \vdash e : \sigma \times \tau}{\Gamma \vdash \mathrm{fst}\, e : \sigma} \qquad \frac{\Gamma \vdash e : \sigma \times \tau}{\Gamma \vdash \mathrm{snd}\, e : \tau}$$

1.3 Monoidal categories

Category theory can be viewed as the mathematicians attempt to turn functions into first-class citicens. Initally, category was based on the category ***set*** of sets and functions. This provided hom-sets for other categories. But often hom-sets carried structure of their own, so the question

arose as to which structure on a category $\mathcal{V}$ was necessary in order for $\mathcal{V}$ to be a viable replacement for ***set***. This lead to "enriched category theory". A very simple example of the usefulness arises by interpreting pre-ordered sets (eqipped with a refelxive and transitive relation) as categories enriched in $\mathbf{2}$, the 2-element chain with $0 < 1$.

Since the original hom-set-driven definition of categories made essential use of the cartesian product $\times$ in ***set*** and its neutral object 1, an obvious strategy was to abstract thier properties that were essential for defining categories.

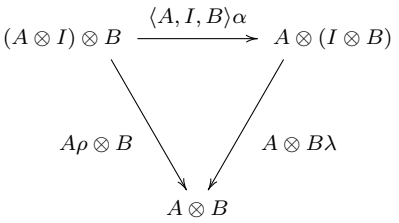
This leads to the consideration of so-called monoidal categories (cf. AAT-notes). For completeness and concistend notation we recall the relevant definitions.

1.3.00 Definition. A *monoidal category* $\langle \mathcal{V}, \otimes, I \rangle$ is a category $\mathcal{V}$ equipped with

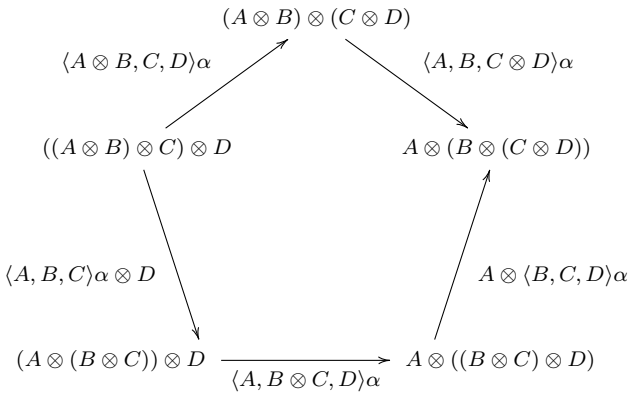
- ▷ a functor $\mathcal{V} \times \mathcal{V} \xrightarrow{\otimes} \mathcal{V}$, called *tensor product*;
- ▷ a so-called *unit object* $I \in \mathcal{V}$;
- ▷ a natural isomorphism α from $\mathcal{V}^3 \xrightarrow{\otimes \times \mathcal{V}} \mathcal{V}^2 \xrightarrow{\otimes} \mathcal{V}$ to $\mathcal{V}^3 \xrightarrow{\mathcal{V} \times \otimes} \mathcal{V}^2 \xrightarrow{\otimes} \mathcal{V}$, as of recently named the *associator*;
- ▷ natural isomorphisms $I \otimes \mathcal{V} \xRightarrow{\lambda} \mathcal{V}$ and $\mathcal{V} \otimes I \xRightarrow{\rho} \mathcal{V}$, nowadays called the *left* and *right unitor*, respectively;

subject to two *coherence conditions*:

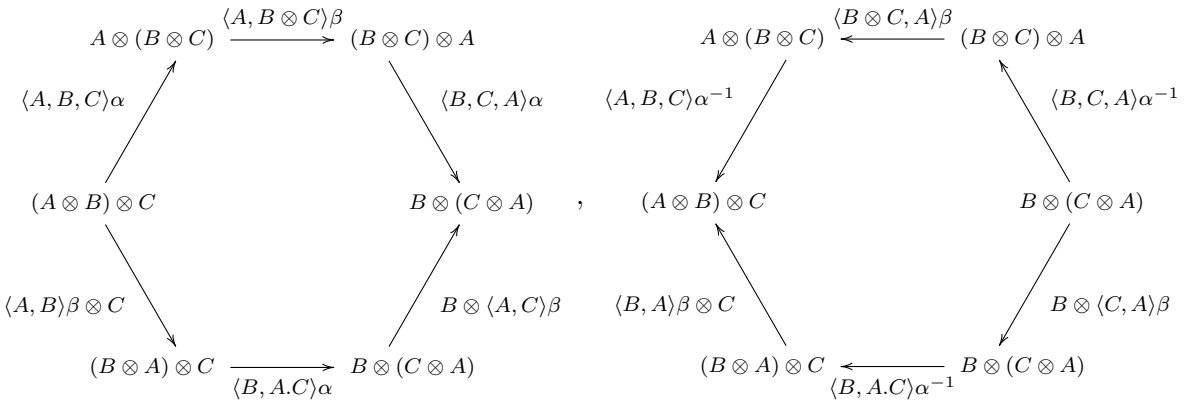
- the *triangle identity*



- the *the pentagon identity*



A monoidal category $\langle \mathcal{V}, \otimes, I \rangle$ is called *braided*, if in addition there is a natural isomorphism $A \otimes B \xrightarrow{\langle A,B\rangle\beta} B \otimes A$, the *braiding*, subject to the following *hexagon identities*



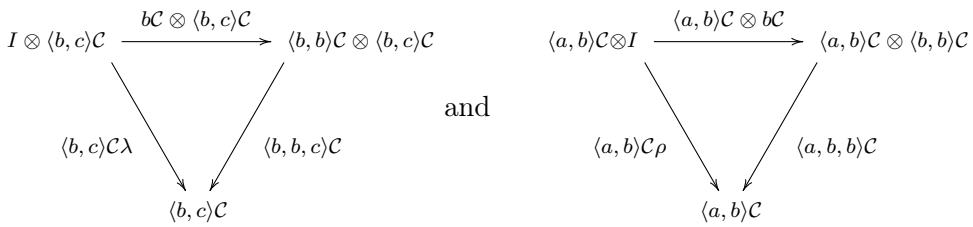
Finally, $\langle \mathcal{V}, \otimes, I \rangle$ is called *symmetric*, if in addition $\langle B, A\rangle\beta = \langle A, B\rangle\beta^{-1}$; in that case one of the hexagons suffices.

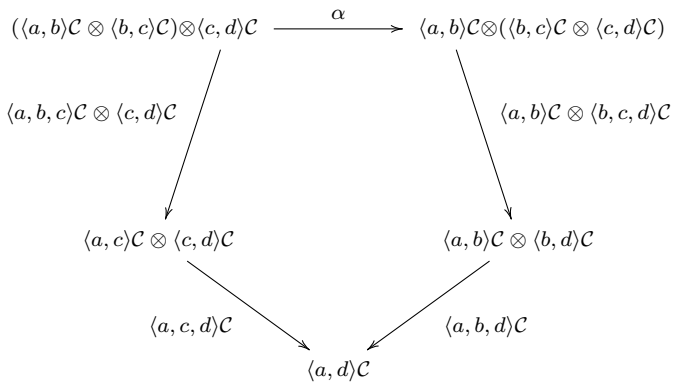
To save space and enhance readability we will often just write α, λ or ρ without the arguments, when those can easily be recovered from the context.

1.3.01 Definition. Given a monoidal category $\langle \mathcal{V}, \otimes, I \rangle$, a $\mathcal{V}$*-category* $\mathcal{C}$ consists of

- ▷ a class C_0 of *objects*;
- ▷ a family $\langle a, b\rangle\mathcal{C}$ of $\mathcal{V}$-objects, called *hom-objects*;
- ▷ a family of $\mathcal{V}$-morphisms $I \xrightarrow{a\mathcal{C}} \langle a, a\rangle\mathcal{C}$;
- ▷ a family of $\mathcal{V}$-morphisms $\langle a, b\rangle\mathcal{C} \otimes \langle b, c\rangle\mathcal{C} \xrightarrow{\langle a,b,c\rangle\mathcal{C}} \langle a, c\rangle\mathcal{C}$

subject to the obvious unit and associativity conditions (note that the number of arguments to $\mathcal{C}$ distinguishes between identities, hom-sets and composition morphisms):





Note that a monoidal category $\langle\mathcal{V},\otimes,I\rangle$ need not be "self-enriched", *i.e.*, it need not be a $\mathcal{V}$-category itself.

But if $\mathcal{V}$ has internal homs, say $A\triangleright B$, we at least expect them to relate to the external homs $\langle A,B\rangle\mathcal{V}$ via

$$\langle I, A\triangleright B\rangle\mathcal{V}\cong\langle A,B\rangle\mathcal{V}$$

Since I is the unit for $\otimes$, one furthermore should expect

$$\langle I, A\triangleright B\triangleright C\rangle\mathcal{V}\cong\langle A, B\triangleright C\rangle\mathcal{V}\cong\langle B\otimes A, C\rangle\mathcal{V}$$

This amounts to $\langle\mathcal{V},\otimes,I$ being "post-closed", *i.e.*, each functor $\mathcal{V}\xrightarrow{B\otimes -}\mathcal{V}$ has a right adjoint $\mathcal{V}\xrightarrow{B\triangleright -}\mathcal{V}$. In particular, if the tensor $\otimes$ is the categorical product $\times$, which in particular is symmetric, we arrive at the notion of "cartesian closedness".

Careful: in the literature often the functor $-\otimes B$ is considered as well; the choice of composition order may come into play here!

Recall from AAT: monoidal categories are 1-object bicategories, and for those we have notions of pre- and post-closeness.

1.4 Closed categories

There is a different path to self-enrichment via closed categories, due to [Eilenberg, Kelly 1965].

1.4.00 Definition. *pre-closed category*: $\mathcal{V}^{\mathrm{op}}\otimes\mathcal{V}\xrightarrow{\triangleright}\mathcal{V}$ with neutral I; *post-closed category*: $\mathcal{V}\otimes\mathcal{V}^{\mathrm{op}}\xrightarrow{\triangleleft}\mathcal{V}$ with neutral J; *symmetry*: $A\triangleright B\triangleright C\cong B\triangleright A\triangleright C$

We cannot express pre/post-closed categories as bicategories with certain properties, as in bicategories we automatically have a composition operation for 1-cells. [Maybe on graphs one can define other binary operations on arrows with comon domain or common codomain instead of the usual composition of arrows???]

1.5 Monoidal closed categories

Work out the connection between the functors $\mathcal{V} \times \mathcal{V} \xrightarrow{\otimes} \mathcal{V}$ and $\mathcal{V}^{\mathrm{op}} \times \mathcal{V} \xrightarrow{\triangleright} \mathcal{V}$. Whenever $B \otimes -$ has a right adjoint $B \triangleright -$ for ecah B, one can construct $\triangleright$ from $\otimes$, and whenever $B \triangleright -$ has a left adjoint $B \otimes -$ for ecah B, one can construct $\otimes$ from $\triangleright$. Thie make essential use of the Australian "mate calculus".

Which conditions do we need on a pre-closed structure $\langle \triangleright, I \rangle$ and a post-closed structure $\langle \triangleleft, I \rangle$ to ensure that both yield the same tensor $\langle \otimes, I \rangle$?

1.6 Why cartesian closedness?

The simply typed λ-calculus would seem to freely generate a pre-closed category. Why ist this symmetric? Need to find λ-terms between $\sigma \to \tau \to \chi$ and $\tau \to \sigma \to \chi$ such that both composites are equivalent to the respective identitites.

2 Monads revisited

2.0 Extension systems

Monads in category theory are usually seen as monoids in the endo-hom-category of some object in a 2-category. But in case of the prototypical 2-category $\boldsymbol{Cat}$, where the notion originated [God58], the iteration of the underlying endo-functor $\boldsymbol{C} \xrightarrow{T} \boldsymbol{C}$ required to even formulate the multiplication $TT \xRightarrow{\mu} T$ may be an "expensive" operation.

For this reason an alternative description of monads has been developed, which as of 2010 goes by the name "extension system" [MW10], but which occurs in Manes' 1976 book [Man76] as "algebraic theory", and in R. F. C. Walters thesis [Wal70] as "full device".

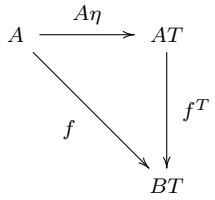
2.0.00 Definition. An *extension system* $\mathcal{T} = \langle T, \eta, (-)^T \rangle$ on a category $\boldsymbol{C}$ consists of

- ▷ a family of distinguished $\boldsymbol{C}$-morphisms $A \xrightarrow{A\eta} AT$ called *Kleisli-units*; $\boldsymbol{C}$-morphisms with codomain of the form AT will be called *Kleisli-morphisms*, while in computer science objects of the Forme AT are often referred to as *computations of type* A;
- ▷ for every Kleisli-morphism $A \xrightarrow{f} BT$ an *extension* $AT \xrightarrow{f^T} BT$ subject to the following axioms:

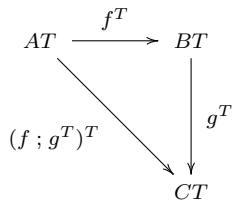
(ES-0) every Kleisli-unit $A \xrightarrow{A\eta} AT$ extends to the $\boldsymbol{C}$-identitity on AT

$$(A\eta)^T = id_{AT}$$

(ES-1) every Kleisli-morphism $A \xrightarrow{f} BT$ can be recovered from its extension by pre-composition with the Kleisli-unit on A



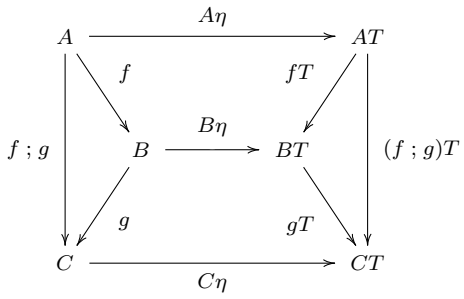
(ES-2) extensions of Kleisli-morphisms are closed under composition in $\boldsymbol{C}$



2.0.01 Theorem. *Extension systems on $\boldsymbol{C}$ are in bijective correspondence with monads on $\boldsymbol{C}$.*

Proof. Starting with an extension system $\mathcal{T} = \langle T, \eta, ()^T \rangle$, we first extend the function T on objects to a functor. Simultaneously we establish that the $\boldsymbol{C}$-morphisms $A\eta$, $A \in \boldsymbol{C}$ comprise a natural transformation.

For $A \xrightarrow{f} B$ define $AT \xrightarrow{fT} BT$ as the extension $(f \,;\, B\eta)^T$. Condition (ES-0) guarantees the preservation of units. To establish the preservation of composition, consider the following diagram for $A \xrightarrow{f} B \xrightarrow{g} C$:

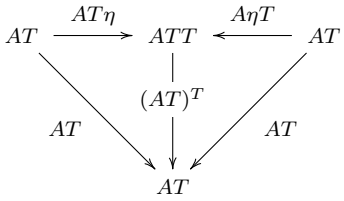


The left triangle commutes by default, while the trapezoids and the outer rectangle commute by definition of fT, gT, and $(f \,;\, g)T$, respectively. In order to show that the right triangle commutes, we observe that $fT \,;\, gT$ as the composition of two extensions $(f \,;\, B\eta)^T$ and $(g \,;\, C\eta)^T$ by (ES-2) is the extension of

$$f \,;\, B\eta \,;\, (g \,;\, C\eta)^T = f \,;\, B\eta \,;\, gT = f \,;\, g \,;\, C\eta$$

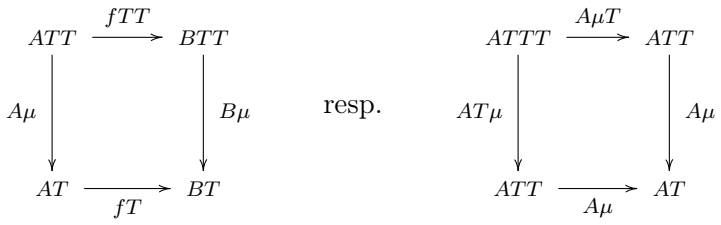
and hence by (ES-1) coincides with $(f\,;g)T$. The naturality of η is also given by the trapezoids above.

We now define the multiplocation $ATT \xrightarrow{A\mu} AT$ to be the extension of the identity on AT. Concerning the unit axioms, the left triangle of



commutes by (ES-1). For the right triangle observe that $A\eta T$ is the extension of $A\eta\,;AT\eta$. Hence the composition with $A\mu = (AT)^T$ is the extension of $A\eta\,;AT\eta\,;(AT)^T = (A\eta)^T = AT$.

The naturality condition for μ and its associativity follow the same pattern. Establishing



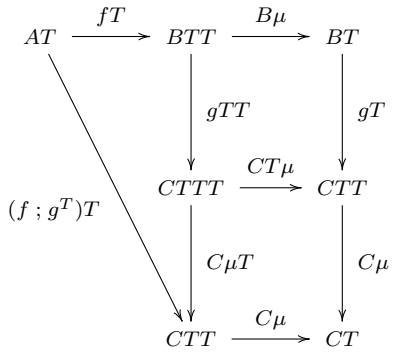
requires expressing all four morphisms as Kleisli-extensions and then using (ES-2) to identify both composites as the extension of the same Kleisli-morphism. On the left we obtain

$$fTT\,;B\mu = \big(fT\,;BT\eta\,;B\mu\big)^T = (fT)^T = \big(AT\,;(f\,;B\eta)^T\big)^T = A\mu\,;fT$$

while on the right we have

$$A\mu T\,;A\mu = \big(A\mu\,;AT\eta\,;A\mu\big)^T = (A\mu)^T = \big(ATT\,;(AT)^T\big)^T = AT\mu\,;A\mu$$

Conversely, starting from a monad $\boldsymbol{T} = \langle T, \eta, \mu\rangle$, we just need to define the extensions of Kleisli-morphisms to obtain a candidate for an extension system. Given $A \xrightarrow{f} BT$, define $AT \xrightarrow{f^T} BT$ to be the composite $AT \xrightarrow{fT} BTT \xrightarrow{B\mu} BT$. Condition (ES-0) then immediately follows from one of the unit requirements of a monad, while (ES-1) is an easy consequence of the naturality of η and the other unit requirement. In order to establish (ES-2), for Kleisli-morphisms $A \xrightarrow{f} BT$ und $B \xrightarrow{g} CT$ consider the following diagrams:



The squares commute by naturality and associativity of μ, while the triangle is the T-image of the definition of $f\,;g^T$. Hence the resulting morphism from AT to CT by definition is $(f\,;g^T)^T$, as required by (ES-2).

It remains to show that the transitions from extension systems to monads and back are mutually inverse. This is clear on the level of T and η. Given an extension system $\mathcal{T} = \langle T, \eta, (-)^T\rangle$ and the resulting monad $\boldsymbol{T} = \langle T, \eta, \mu\rangle$, consider the extension system $\mathcal{T}' = \langle T, \eta, (-)^\circ\rangle$. The new extension of a Kleisli-morphism $A \xrightarrow{f} BT$ by (ES-2) of the original extension system is given by

$$fT\,;B\mu = (f\,;B\eta)^T\,;(BT)^T = (f\,;B\eta\,;B\mu)^T = f^T$$

and hence coincides with the original extension.

Conversely, for a monad $\boldsymbol{T} = \langle T, \eta, \mu\rangle$ and the resulting extension system $\mathcal{T} = \langle T, \eta, (-)^T\rangle$, consider the new monad $\boldsymbol{T}' = \langle\, T, \eta, \mu'\rangle$ derived from this. The new multiplication satisfies

$$A\mu' = (AT)^T = ATT\,;A\mu = A\mu$$

and hence coincides with the original one. □

As our terminology suggests, the Kleisli-category of a monad $\boldsymbol{T} = \langle T, \eta, \mu\rangle$ has a particularly simple description in terms of the corresponding extension system $\mathcal{T} = \langle T, \eta, (-)^T\rangle$:

2.0.02 Definition. The Kleisli-category $\boldsymbol{C_T}$ has

- ▷ the same objects as $\boldsymbol{C}$;
- ▷ the Kleisli-morphisms into AT as morphisms into A

$$\langle B, A\rangle\,\boldsymbol{C_T} = \langle B, AT\rangle\,\boldsymbol{C}$$

- ▷ a composition by means of the $\boldsymbol{C}$-composition with the extension of the second argument, *cf.* axiom (ES-2):

$$\langle C, BT\rangle\,\boldsymbol{C} \times \langle B, AT\rangle\,\boldsymbol{C} \xrightarrow{\langle C, B, A\rangle\,\boldsymbol{C_T}} \langle C, AT\rangle\,\boldsymbol{C}$$

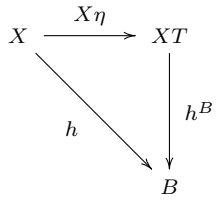
maps $\langle C \xrightarrow{g} BT, B \xrightarrow{f} AT\rangle$ to $C \xrightarrow{g} BT \xrightarrow{f^T} AT$.

On the other hand, the Eilenberg-Moore category of a monad $\boldsymbol{T}$ has a rather more complicated description in these terms.

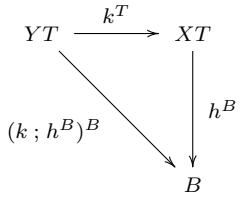
2.0.03 Definition. A $\mathcal{T}$-algebra consists of

- ▷ a $\boldsymbol{C}$-object B;
- ▷ for every $\boldsymbol{C}$-morphism $X \xrightarrow{h} B$ a distinguished extension $XT \xrightarrow{h^B} B$ called *structure-morphism* subject to

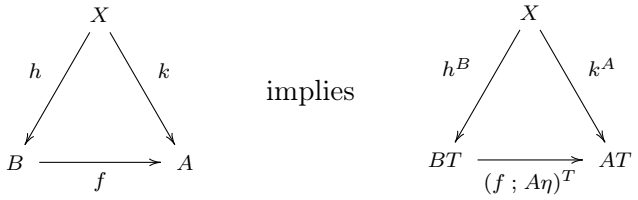
(TA-0) h can be recovered from h^B by pre-composition with $X\eta$



(TA-1) structure-morphisms are stable under pre-composition with extended Kleisli-morphisms:



A $\mathcal{T}$-*homomorphism* $B \xrightarrow{f} A$ between $\mathcal{T}$-algebras has to commute with all structure morphisms:

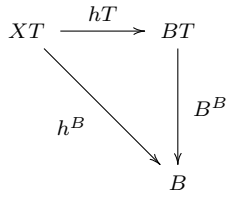


2.0.04 Remark. Unfortunately, depending on the size of $\boldsymbol{C}$, a $\mathcal{T}$-algebra my be specified by a proper class of structure morphisms, which certainly is undesirable. As will be shown below, a single structure morphism, namely $(\mathit{id}_B)^B$, suffices to specify the algebra structure. But then the notion of $\mathcal{T}$-morphism will require adjustment.

Mikołaj Bojańczyk in his treatment of monoids and formal languages does use epimorphic structure morphisms of the form $XT \xrightarrow{h} BT$ to turn a set B into a monoid..

2.0.05 Proposition. *There is a bijective correspondence between $\mathcal{T}$-algabras and Eilenberg-Moore algebras for the monad $\boldsymbol{T}$.*

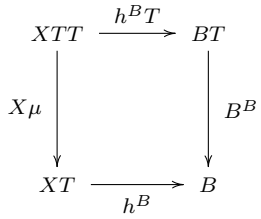
Proof. First notice that by the distinguished extensions h^B for $X \xrightarrow{h} B$ are determined by pre-composing hT with $BT \xrightarrow{B^B} B$:



(2.0-00)

By definition, $hT := (h\,;B\eta)^T$, hence by (TA-1) the diagonal morphism is $(h\,;B\eta\,;B^B)^B$, which by (TA-0) reduces to h^B.

In order to estblish $\langle B, B^B\rangle$ as an EM-algebra, we only need to show (EM-1), since by (TA-0) we already know that B^B is right inverse to $B\eta$. In fact, we show a bit more than (EM-1), namely for $X \xrightarrow{h} B$ we have



(2.0-01)

Utilizing (TA-1) for both composites we obtain

$$\begin{aligned} h^BT\,;B^B &= \big(h^B\,;B\eta\big)^T\,;B^B = \big(h^B\,;B\eta\,;B^B\big)^T \\ &= \big(h^B\big)^T = \big(XT\,;h^B\big)^T = (XT)^T\,;h^B = X\mu\,;h^B \end{aligned}$$

Conversely, if $\langle B,\xi\rangle$ is an EM-algebra, setting $B^B := \xi$ and defining h^B by Diagram 2.0-00 condition (TA-0) for $X \xrightarrow{h} B$ follows from

$$\begin{aligned} X\eta\,;h^B &= X\eta\,;hT\,;B^B \\ &= X\eta\,;(h\,;B\eta)^T\,;B^B = h\,;B\eta\,;B^B = h \end{aligned}$$

On the other hand, for $Y \xrightarrow{k} XT$ Condition (TA-1) is a consequence of

$$k^T\,;h^B = kT\,;X\mu\,;h^B = kT\,;h^BT\,;B^B = (k\,;h^B)T\,;B^B = \big(k\,;h^B\big)^B$$

References

[God58] R. Godement. *Théorie des faisceaux*. Hermann, Paris, 1958.

[Man76] E. G. Manes. *Algebraic Theories*. Graduate Texts in Mathematics. Springer, 1976.

[MW10] F. Marmolejo and R. J. Wood. Monads as extension systems - no iteration is necessary. *Theory and Applications of Categories*, 24(4):84–113, 2010.

[Wal70] R. F. C. Walters. *A categorical approach to universal algebra*. PhD thesis, The Australian Natinal University, Canberra, Jan 1970.

Index

Jürgen Koslowski (koslowj@tu-bs.de)
Theoretical Computer Science
TU Braunschweig
Mühlenpfordstr. 23
D-38106 Braunschweig
Germany