Contents

List of Figures

Notes on type-free vs simply typed lambda calculi and their categorical semantics

Cameron Smith[1,*], Ximo Pechuan[1]
1 Department of Systems and Computational Biology, Albert Einstein College of Medicine, Bronx, NY, USA
∗ **E-mail: cameron.smith@med.einstein.yu.edu**

Todo list

Simply typed lambda calculus

The simply typed lambda calculus (λ-calculus) is essentially equivalent to a cartesian closed category in a sense we can make precise. Both are languages in which it is possible to express transformations via functions between objects with operations that express pairing, projection, and application.

The simply typed lambda calculus is made up of types, terms and equations as follows:

1. Types:

$$\begin{aligned}&\text{basic types: } A, B, \ldots\\ &\text{product types: } A \times B, \ldots\\ &\text{function types: } A \to B, \ldots\end{aligned}$$

2. Terms:

$$\begin{aligned}&\text{variables: } x, y, z, \ldots : A \text{ for each type } A,\\ &\text{constants: } a\colon A, b\colon B, \text{ for each type,}\\ &\text{products: } \langle a, b\rangle\colon A \times B \text{ for } a : A \text{ and } b : B,\\ &\text{first projection: } \mathrm{fst}(c)\colon A \text{ for } c\colon A \times B,\\ &\text{second projection: } \mathrm{snd}(c)\colon A \text{ for } c\colon A \times B,\\ &\text{application: } ca\colon B \text{ for } c\colon A \to B \text{ and } a\colon A,\\ &\text{abstraction: } \lambda x.b\colon A \to B \text{ for } x\colon A \text{ and } b\colon B\end{aligned}$$

3. Equations:

$$\begin{aligned}\mathrm{fst}(\langle a, b\rangle) &= a\\ \mathrm{snd}(\langle a, b\rangle) &= b\\ \langle \mathrm{fst}(c), \mathrm{snd}(c)\rangle &= c\\ (\lambda x.b)a &= b[a/x]\\ \lambda x.cx &= c, \ x \text{ not in } c\\ \lambda x.b &= \lambda y.b[y/x], \ \text{no } y \text{ in } b\end{aligned}$$

A variable reference x is said to be *bound* if it is inside of an abstraction binding x. For example $\lambda x.x$ is a term in which x is bound whereas $\lambda y.xy$ is a term in which x is unbound or *free* and y is bound. A term is said to be *closed* if there are no unbound variables. Therefore, $\lambda x.x$ is a closed term whereas $\lambda y.xy$ is not. Terms are identified according to the equivalence relation $a \sim b$ generated by the equations above.

Categorical semantics for the simply typed lambda calculus

A category can be associated to any particular instantiation of the lambda calculus with given types, variables and constants. To any such language $\mathcal{L}$ satisfying the defining equations of the simply typed lambda calculus, a cartesian closed category of types $\mathcal{C}(\mathcal{L})$ is determined by the following identifications

1. objects: types
2. morphisms: terms $c\colon A \to B$ that are identified for $c \sim c'$ given by the equivalence relation determined by the defining equations of λ-calculus. Two equivalence classes of terms $[a], [b]$ may be identified $[a] = [b]$ if and only if the terms they represent are equivalent, which is to say that $\mathcal{L} \vdash a = b$.
3. identities: $1_A = \lambda x.x$ where $x\colon A$
4. morphism composition: $c \circ b = \lambda x.c(bx)$

The unit laws, associativity of composition, the existence of products, terminal objects, and exponential objects can be verified from this set of identifications to prove that the category $\mathcal{C}(\mathcal{L})$ is cartesian closed.

If we refer to the set of basic types, terms, and equations as a *theory*, $\mathcal{L}$, in the λ-calculus, then $\mathcal{C}(\mathcal{L})$ is the cartesian closed category presented by the generators (types and terms) and relations (equations) stated by the λ-calculus over $\mathcal{L}$. A model of a such a theory $\mathcal{L}$ in the λ-calculus in a cartesian closed category $\mathcal{C}$ is given by an assignment of types and terms in $\mathcal{L}$ to objects and morhphisms in $\mathcal{C}$:

$$\begin{aligned} X \text{ basic type } &\rightsquigarrow [\![X]\!] \\ b\colon A \to B \text{ basic term } &\rightsquigarrow [\![b]\!]\colon [\![A]\!] \to [\![B]\!] \end{aligned}$$

which can be naturally extended to the other types and terms that are built upon these. Moreover the equations associated to the theory $\mathcal{L}$ are required to be satisfied so that:

$$\mathcal{L} \vdash [a] = [b]\colon A \to B \Longrightarrow [\![a]\!] = [\![b]\!]\colon [\![A]\!] \to [\![B]\!]$$

This semantic association between a theory $\mathcal{L}$ in λ-calculus and models of such theories in cartesian closed categories is referred to as *denotational semantics* for the λ-calculus. Indeed, there is a logical completeness theorem associated to such semantics stating that for any theory $\mathcal{L}$ in the λ-calculus equivalence of terms is mirrored precisely by equivalence of the semantic interpretations of those terms in all models for $\mathcal{L}$ in cartesian closed categories. Although we will not expose the proof of this theorem here, it is important to note that it is not sufficient to consider only the category **Sets** as then there are some equations that hold among the models in that restricted context that cannot be proved in a theory of λ-calculus.

The type-free lambda calculus and its categorical semantics

The type-free lambda calculus can be seen as a special case of the simply-typed lambda calculus in which the set of basic types is restricted to contain only a single type. With the typing constraint imposed by

having multiple types removed, it should be possible to perform self-application as in the term $\lambda x.xx$. If the second occurence of x in this term has type D and the whole term xx has type D then the first occurrence of x in the term xx must be construable as having type $[D \to D]$. A presentation of the type-free theory, $\mathbb{T}$, can then be given in a manner analogous to that of the simply typed lambda calculus [1]:

1. Types:

$$\text{one basic type: } D$$

2. Terms:

$$\begin{aligned}&\text{variables: } x, y, z, \ldots : D\\ &\text{constants: } i\colon [D \to D] \to D,\ r\colon D \to [D \to D]\end{aligned}$$

3. Equations:

$$\begin{aligned}\lambda x.ri(x) &= \lambda x.x\\ \lambda x.ir(x) &= \lambda x.x\end{aligned}$$

On this basis, we are led to the requirement that

$$[D \to D] \cong D,$$

which is a particular instantiation of a more general phenomenon referred to as a recursive domain equation. The process by which $[D \to D]$ is formed in the first place is via the internal Hom bifunctor $F \equiv [-,-]\colon \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$ on a category $\mathcal{C}$, which thereby restricts consideration for the purpose of finding denotational semantics for the type-free λ-calculus to categories $\mathcal{C}$ that are closed since this is a necessary condition for it to posses the necessary structure to define the internal Hom bifunctor. In this light, the solution D to the equation $[D, D] \cong D$ is a fixed point $F(D, D) \cong D$ of the internal Hom bifunctor on some category $\mathcal{C}$.

Given these restrictions, it is obvious that it will not be sufficient in the search for non-trivial models of the type-free theory of the λ-calculus to restrict consideration to the category **Sets** because in such a category F has a fixed point D only when D is the one element set: $D = \{*\}$. However, $\mathbb{T}$ does not necessarily prove $\lambda y.ir(y) = \lambda y.y$, which would be true for $D = \{*\}$. Because something provable in **Sets** is not necessarily provable in the type-free λ-calculus there is no sense in which semantics in the category **Sets** could be complete with respect to the type-free λ-calculus.

We will very briefly sketch the solution to this particular problem and refer to other sources for caveats and details [2–6]. In order to remedy this situation Scott eventually introduced *domains*, which can be viewed as *continuous lattices* (this construction appears with minor modifications with respect to the purpose of the present exposition in terms of various other closely related kinds of objects and their associated categories such as complete partial orders and complete lattices among others). A continuous lattice is a complete lattice D such that

$$\forall d \in D, d = \bigvee \left\{ \bigwedge U | d \in U, U \text{ Scott-open}, U \subseteq D \right\}$$

where *Scott-open* refers to a subset U of a domain D that is upward closed in the sense that if $\bigvee \Delta \in U$ for a directed subset $\Delta \subseteq D$ then $U \cap \Delta \neq \emptyset$ and Scott-continuous functions are required to be monotonic

Include appropriate definitions from lattice theory (e.g. lattice, meet, join, completeness, etc.)

and preserve least upper bounds of directed subsets of their domains. These continuous lattices can be equivalently characterized in topological terms as T_0-spaces in which every continuous function $f\colon P \to D$ from a subspace $P \subseteq S$ can be extended to a Scott-continuous function $\hat{f}\colon S \to D$.

Domains of this form taken as objects and Scott-continuous functions between them taken as morphisms form cartesian closed categories where the internal-hom (i.e. exponential) objects were naturally themselves not sets but complete lattices of Scott-continuous functions. Since these form a cartesian closed category they could be used as described above to provide semantics for the simply-typed λ-calculus. However, categories of domains can also be constructed that have reflexive objects, D_∞, which solve the recursive domain equation $F(D_\infty, D_\infty) = D_\infty$ that arises naturally as described above in the analysis of the type-free λ-calculus since, there, objects serve both as arguments and as functions that can be applied to such arguments.

The way in which such solutions D_∞ are constructed is via a generalization of the least fixed-point of a continuous function. A continous function $f\colon D \to D$ may have fixed-points x such that $f(x) = x$. The least fixed-point for such a continuous function f is then given by

$$fix(f) = \bigvee_{n\in\omega} f^n(\bot)$$

Now, we would like to extend this notion to the internal-hom bifunctor $F \equiv [-,-]\colon \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$ for $\mathcal{C}$ a category of continuous lattices and Scott-continuous functions. Given domains D and E as objects in such a category, we can define an *embedding-projection pair* from D to E as a pair of Scott-continuous functions $i\colon D \to E$ and $r\colon E \to D$ such that $r \circ i = id_D$ and $i \circ r \leq id_E$ where the relation of order on functions is given pointwise. Reflexive domains are given by inverse limits (or projective limits, which are specializations of limits, as opposed to colimits, in category theory) of sequences of projections r_n given by:

$$\begin{aligned}
&D_0 = \text{ some object, } D, \text{ in } \mathcal{C},\\
&D_1 = [D_0 \to D_0],\\
&D_{n+1} = [D_n \to D_n],\\
&(r_n\colon D_{n+1} \to D_n)_{n\in\omega}.
\end{aligned}$$

The fact that a suitable choice for the initial embedding-projection pair $\langle i_0, r_0\rangle$ ultimately requires that the limit of the above sequence of projections coincide with the colimit of the sequence of embeddings

$$(i_n\colon D_n \to D_{n+1})_{n\in\omega}.$$

It is eventually possible to obtain

$$\lim_{\leftarrow}(D_n, r_n) \cong D_\infty \cong \lim_{\rightarrow}(D_n, i_n).$$

and this particular construction of D_∞ provides a fixed-point $\mathrm{FIX}(F)$ so that $F(D_\infty, D_\infty) = D_\infty$ solves the given recursive domain equation associated to the untyped λ-calculus. This allows us to interpret the terms of the untyped lambda calculus as being the objects and morphisms D_∞ and $[D_\infty, D_\infty]$ respectively, which are now seen to be one and the same since $D_\infty \cong D_\infty^{D_\infty}$.

Acknowledgments

The authors would like to thank Jay Sulzberger and Noson Yanofsky for helpful discussions.

References

1. Awodey S (2000) Topological representation of the λ-calculus. Math Struct Comput Sci 10: 81–96.
2. Barendregt H (1985) The Lambda Calculus, Its Syntax and Semantics. North Holland, 621 pp.
3. Smyth MB, Plotkin GD (1982) The Category-Theoretic Solution of Recursive Domain Equations. SIAM J Comput 11: 761–783.
4. Freyd P (1990) Algebraically Complete Categories. In: Carboni A, Pedicchio MC, Rosolini G, editors, Categ. Theory. Como.
5. Abramsky S, Jung A (1995) Domain Theory. In: Abramsky S, Gabbay DM, Maibaum TSE, editors, Handb. Log. Comput. Sci. Vol. 3. Semant. Struct., Oxford University Press, volume 3, chapter 3. pp. 1–168.
6. Cattani GL, Fiore MP (2007) The Bicategory-Theoretic Solution of Recursive Domain Equations. Electron Notes Theor Comput Sci 172: 203–222.

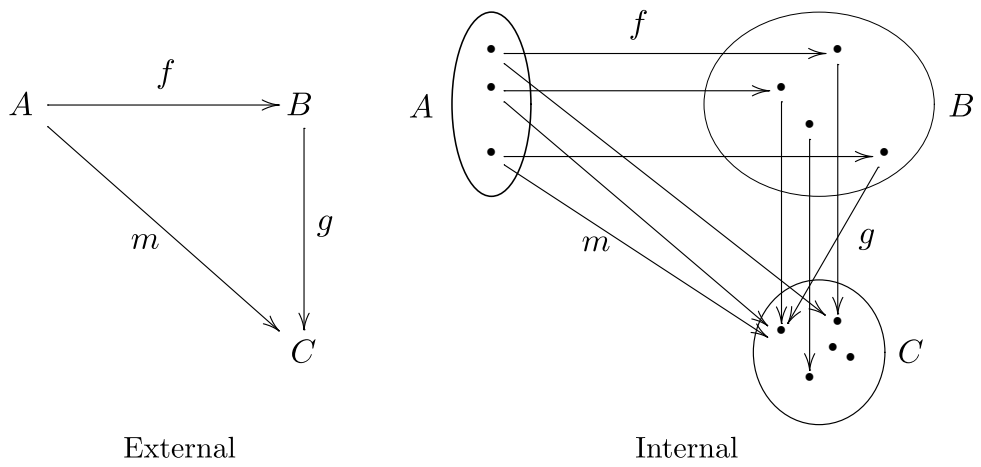


Figure 1. In the external diagram on the left one simply considers *objects* such as A,B, and C and morphisms between them f, g, and m. The internal perspective, which in this case is specific to the category of **Sets**, demonstrates the particular manner in which the equation $g \circ f = m$ may be satisfied in terms of the internal structure of the objects A, B, and C. Of course, in this case there are many other ways of satisfying the given equation; however, there are also questions that can be answered about the relationship between A, B, and C given either $g \circ f = m$ or $g \circ f \neq m$ without necessarily knowing the particular manner in which either equation is satisfied by the relationships among the internal structures of the objects involved.

Background

Categories

An intuitive picture to which one can anchor intuition when thinking in category theoretic terms is presented in Fig. 1. In category theory, one can always consider the duality between the intrinsic structure of *objects* and the manner in which that intrinsic structure can be expressed externally in terms of patterns of constraints placed upon the relationships between *morphisms* among objects.

A *category* $\mathcal{C}$ consists of the following data:

1. a class $\mathrm{ob}(\mathcal{C})$ of objects (of $\mathcal{C}$)
2. for each ordered pair (A, B) of objects of $\mathcal{C}$, a collection (we will assume it is a set) $\hom(A, B)$ of morphisms from the domain A to the codomain B
3. a function $\circ : \hom(A, B) \times \hom(B, C) \to \hom(A, C)$ called composition.

Some of this section is "textbook" material taken from planetmath.org. It is a helpful reference to definitions for reviewers.

We normally denote $\circ(f, g)$ by $g \circ f$ for morphisms f, g. The above data must satisfy the following axioms: for objects A, B, C, D,

A1: $\hom(A, B) \cap \hom(C, D) = \emptyset$ whenever $(A, B) \neq (C, D)$, i.e. the intersection is non-trivial only when $A = C$ and $B = D$.

A2: (Associativity) if $f \in \hom(A, B)$, $g \in \hom(B, C)$ and $h \in \hom(C, D)$, $h \circ (g \circ f) = (h \circ g) \circ f$

A3: (Existence of an identity morphism) for each object A there exists an identity morphism $id_A \in \hom(A, A)$ such that for every $f \in \hom(A, B)$, $f \circ id_A = f$ and $id_A \circ g = g$ for every $g \in \hom(B, A)$.

Some examples of categories:

- **0** is the empty category with no objects or morphisms, **1** is the category with one object and one (identity) morphism.
- If we assume we have a universe U which contains all sets encountered in "everyday" mathematics, **Set** is the category of all such small sets with morphisms being set functions
- **Top** is the category of all small topological spaces with morphisms continuous functions
- **Grp** is the category of all small groups whose morphisms are group homomorphisms

Remark. If $\hom(A, B)$ in the second condition above is not required to be a set (but a class), we usually call $\mathcal{C}$ a *large category*.

Initial and terminal objects

An *initial object* in a category $\mathcal{C}$ is an object A in $\mathcal{C}$ such that, for every object X in $\mathcal{C}$, there is exactly one morphism $A \longrightarrow X$. A *terminal object* in a category $\mathcal{C}$ is an object B in $\mathcal{C}$ such that, for every object X in $\mathcal{C}$, there is exactly one morphism $X \longrightarrow B$. A *zero object* in a category $\mathcal{C}$ is an object 0 that is both an initial object and a terminal object. All initial objects (respectively, terminal objects, and zero objects), if they exist, are isomorphic in $\mathcal{C}$.

Functors

Given two categories $\mathcal{C}$ and $\mathcal{D}$, a covariant *functor* $T : \mathcal{C} \to \mathcal{D}$ consists of an assignment for each object X of $\mathcal{C}$ an object $T(X)$ of $\mathcal{D}$ (i.e. a "function" $T : \mathrm{Ob}(\mathcal{C}) \to \mathrm{Ob}(\mathcal{D})$) together with an assignment for every morphism $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$, to a morphism $T(f) \in \mathrm{Hom}_{\mathcal{D}}(T(A), T(B))$, such that:

- $T(1_A) = 1_{T(A)}$ where 1_X denotes the identity morphism on the object X (in the respective category).
- $T(g \circ f) = T(g) \circ T(f)$, whenever the composition $g \circ f$ is defined.

A contravariant functor $T : \mathcal{C} \to \mathcal{D}$ is just a covariant functor $T : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}$ from the opposite category. In other words, the assignment reverses the direction of maps. If $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$, then $T(f) \in \mathrm{Hom}_{\mathcal{D}}(T(B), T(A))$ and $T(g \circ f) = T(f) \circ T(g)$ whenever the composition is defined (the domain of g is the same as the codomain of f).

Given a category $\mathcal{C}$ and an object X we always have the functor $T : \mathcal{C} \to \mathbf{Sets}$ to the category of sets defined on objects by $T(A) = \mathrm{Hom}(X, A)$. If $f : A \to B$ is a morphism of $\mathcal{C}$, then we define $T(f) : \mathrm{Hom}(X, A) \to \mathrm{Hom}(X, B)$ by $g \mapsto f \circ g$. This is a covariant functor, denoted by $\mathrm{Hom}(X, -)$.

Similarly, one can define a contravariant functor $\mathrm{Hom}(-, X) : \mathcal{C} \to \mathbf{Sets}$.

Natural transformations

Let $\mathcal{C}$ and $\mathcal{D}$ be categories, and let $S, T : \mathcal{C} \to \mathcal{D}$ be covariant functors. Then suppose that for every object A in $\mathcal{C}$ one has a morphism $\eta_A : S(A) \to T(A)$ in $\mathcal{D}$ such that for every morphism $\alpha : A \to B$ in $\mathcal{C}$ the following

$$\begin{array}{ccc} S(A) & \xrightarrow{\eta_A} & T(A) \\ {\scriptstyle S(\alpha)}\downarrow & & \downarrow{\scriptstyle T(\alpha)} \\ S(B) & \xrightarrow{\eta_B} & T(B) \end{array}$$

is commutative. Then we variously write

$$\eta : S \dot{\to} T \quad \text{or} \quad \eta : S \Rightarrow T \quad \text{or} \quad \eta : S \to T$$

and call η a *natural trasformation* from S to T.

One may think of a natural transformation $\eta : S \to T$ as a 'function' from the class of objects of $\mathcal{C}$ to the class of morphisms of $\mathcal{D}$.

As a first example, for every functor $S : \mathcal{C} \to \mathcal{D}$, we can associate the natural transformation $1_S : S \to S$ (the *identity natural transformation* on S) that assigns every object A of $\mathcal{C}$, the corresponding identity morphism $1_{S(A)}$.

Natural transformations are composed in a similar manner to morphisms, but they are nevertheless defined as correspondences between both objects and morphisms as shown in the square commutative diagram depicted above.

More precisely, given three functors $R, S, T : \mathcal{C} \to \mathcal{D}$, and two natural transformations, $\tau : R \to S$ and $\eta : S \to T$, we define the composition of τ with η, written $\eta \bullet \tau$, as a class of morphisms in $\mathcal{D}$ given by

$$(\eta \bullet \tau)_A := \eta_A \circ \tau_A,$$

for every object A in $\mathcal{C}$. It is easy to see that $\eta \bullet \tau$ is a natural transformation, since we may "compose" two commutative squares and obtain a third one:

$$\begin{array}{ccccc} R(A) & \xrightarrow{\tau_A} & S(A) & \xrightarrow{\eta_A} & T(A) \\ \downarrow{\scriptstyle R(\alpha)} & & \downarrow{\scriptstyle S(\alpha)} & & \downarrow{\scriptstyle T(\alpha)} \\ R(B) & \xrightarrow{\tau_B} & S(B) & \xrightarrow{\eta_B} & T(B) \end{array} \quad = \quad \begin{array}{ccc} R(A) & \xrightarrow{\eta_A \circ \tau_A} & T(A) \\ \downarrow{\scriptstyle R(\alpha)} & & \downarrow{\scriptstyle T(\alpha)} \\ R(B) & \xrightarrow{\eta_B \circ \tau_B} & T(B) \end{array}$$

It is easy to see that the composition "operation" on natural transformations is associative:

$$(\zeta \bullet \eta) \bullet \tau = \zeta \bullet (\eta \bullet \tau)$$

for natural transformations $\tau : R \to S$, $\eta : S \to T$, and $\zeta : T \to U$. In addition, any identity natural transformation acts as a compositional identity: if $\tau : R \to S$ and $\eta : S \to T$, then

$$1_S \bullet \tau = \tau \qquad \text{and} \qquad \eta \bullet 1_S = \eta.$$

Adjoint functors

Let $\mathcal{C}$ and $\mathcal{D}$ be (small) categories, and let $T : \mathcal{C} \to \mathcal{D}$ and $S : \mathcal{D} \to \mathcal{C}$ be covariant functors. T is said to be a *left adjoint functor* to S (equivalently, S is a *right adjoint functor* to T) if there is a natural equivalence

$$\nu \colon \operatorname{Hom}_{\mathcal{D}}(T(-), -) \dot{\longrightarrow} \operatorname{Hom}_{\mathcal{C}}(-, S(-)).$$

Here the functor $\operatorname{Hom}_{\mathcal{D}}(T(-), -)$ is a bifunctor $\mathcal{C} \times \mathcal{D} \to \mathbf{Set}$ which is contravariant in the first variable, is covariant in the second variable, and sends an object (C, D) to $\operatorname{Hom}_{\mathcal{D}}(T(C), D)$. The functor $\operatorname{Hom}_{\mathcal{C}}(-, S(-))$ is defined analogously.

This definition needs additional explanation. Essentially, it says that for every object C in $\mathcal{C}$ and every object D in $\mathcal{D}$ there is a function

$$\nu_{C,D} \colon \operatorname{Hom}_{\mathcal{D}}(T(C), D) \xrightarrow{\sim} \operatorname{Hom}_{\mathcal{C}}(C, S(D))$$

which is a natural bijection of hom-sets. Naturality means that if $f\colon C' \to C$ is a morphism in $\mathcal{C}$ and $g\colon D \to D'$ is a morphism in $\mathcal{D}$, then the diagram

$$\begin{array}{ccc} \mathrm{Hom}_{\mathcal{D}}(T(C), D) & \xrightarrow{\nu_{C,D}} & \mathrm{Hom}_{\mathcal{C}}(C, S(D)) \\ \downarrow{\scriptstyle (Tf,g)} & & \downarrow{\scriptstyle (f,Sg)} \\ \mathrm{Hom}_{\mathcal{D}}(T(C'), D') & \xrightarrow{\nu_{C',D'}} & \mathrm{Hom}_{\mathcal{C}}(C', S(D')) \end{array}$$

is a commutative diagram. If we pick any $h : T(C) \to D$, then we have the equation

$$Sg \circ \nu_{C,D}(h) \circ f = \nu_{C',D'}(g \circ h \circ Tf).$$

If $T : \mathcal{C} \to \mathcal{D}$ is a left adjoint of $S : \mathcal{D} \to \mathcal{C}$, then we say that the ordered pair (T, S) is an *adjoint pair*, and the ordered triple (T, S, ν) an *adjunction* from $\mathcal{C}$ to $\mathcal{D}$, written

$$(T, S, \nu) : \mathcal{C} \to \mathcal{D},$$

where ν is the natural equivalence defined above.

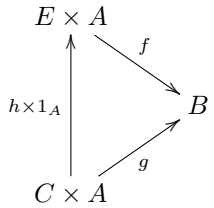
An adjoint to a functor is in some ways like an inverse (as in the case of an adjoint matrix); often formal properties about a functor lead to formal properties of its adjoint (for example the right adjoint to a left-exact functor takes injectives to injectives). An adjoint to any functor is unique up to natural isomorphism.

Exponential objects

Let A, B be objects in a category with finite products $\mathcal{C}$. An object E in $\mathcal{C}$ is called an *exponential object* from A to B if it satisfies the following conditions:

- there is a morphism $f : E \times A \to B$, called an *evaluation morphism*
- for any morphism $g : C \times A \to B$, there is a unique morphism $h : C \to E$ such that $f \circ (h \times 1_A) = g$, where $h \times 1_A : C \times A \to E \times A$ is the product morphism of h and the identity morphism on A.

The two conditions can be summarized by the following commutative diagram:



where h is uniquely determined by g. It is easy to see that any two exponential objects from A to B are isomorphic, hence the existence of an exponential object between two objects is a universal property. We may write $B^A (\cong E$ above) *the* exponential object from A to B.

For example, in the category of sets, **Set**, where products exist between pairs of objects (sets), the exponential from A to B is the set B^A, which is defined as the set of all functions from A to B. The evaluation morphism is the function $ev : B^A \times A \to B$ given by $ev(f, a) = f(a)$, where $f \in B^A$ and $a \in A$. If $g : C \times A \to B$ is any function, then we define $h : C \to B^A$ by $h(c)(a) = g(c, a)$. Then

$ev \circ (h \times 1_A)(c, a) = ev(h(c), a) = h(c)(a) = g(c, a)$, and ev is universal (in the sense of the second condition above).

Since each h is uniquely determined by g in the above definition, and conversely every h determines a g by the formula $g = f \circ (h \times 1_A)$, we have a bijection

$$\hom(C \times A, B) \cong \hom(C, B^A).$$

If an exponential object exists between every pair of objects in category C with finite products, then we say that C *has exponentials*. According to the bijection above, we see that the functor $\cdot \times A : \mathcal{C} \to \mathcal{C}$ has a right adjoint, namely $\cdot^A : \mathcal{C} \to \mathcal{C}$, called the *exponential functor*.

Cartesian closed category

A category $\mathcal{C}$ with finite products is said to be *Cartesian closed* if each of the following functors has a right adjoint

1. $\mathbf{0} : \mathcal{C} \to \mathbf{1}$, where $\mathbf{1}$ is the trivial category with one object 0, and $\mathbf{0}(A) = 0$
2. the diagonal functor $\delta : \mathcal{C} \to \mathcal{C} \times \mathcal{C}$, where $\delta(A) = (A, A)$, and
3. for any object B, the functor $(- \times B) : \mathcal{C} \to \mathcal{C}$, where $(- \times B)(A) = A \times B$, the product of A and B.

Furthermore, we require that the corresponding right adjoints for these functors to be

1. any functor $\mathbf{1} \to \mathcal{C}$, where 0 is mapped to an object T in $\mathcal{C}$. T is necessarily a terminal object of $\mathcal{C}$.
2. the product (bifunctor) $(- \times -) : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ given by $(- \times -)(A, B) \mapsto A \times B$, the product of A and B.
3. for any object B, the exponential functor $(-^B) : \mathcal{C} \to \mathcal{C}$ given by $(-^B)(A) = A^B$, the exponential object from B to A.

In other words, a Cartesian closed category $\mathcal{C}$ is a category with finite products, has a terminal objects, and has exponentials. It can be shown that a Cartesian closed category is the same as a finitely complete category having exponentials.

Examples of Cartesian closed categories are the category of sets **Set** (terminal object: any singleton; product: any Cartesian product of a finite number of sets; exponential object: the set of functions from one set to another) the category of small categories **Cat** (terminal object: any trivial category; product object: any finite product of categores; exponential object: any functor category), and every elementary topos.

Monoidal categories

A *monoidal category* is a category which has the structure of a monoid, that is, among the objects there is a binary operation which is associative and has an unique neutral or unit element. Specifically, a category $\mathcal{C}$ is *monoidal* if

1. there is a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$, where the images of object (A, B) and morphism (f, g) are written $A \otimes B$ and $f \otimes g$ respectively,
2. there is an isomorphism $a_{ABC} : (A \otimes B) \otimes C \cong A \otimes (B \otimes C)$, for arbitrary objects A, B, C in $\mathcal{C}$, such that a_{ABC} is natural in A, B and C. In other words,

- $a_{-BC} : (- \otimes B) \otimes C \Rightarrow - \otimes (B \otimes C)$ is a natural transformation for arbitrary objects B, C in $\mathcal{C}$,
- $a_{A-C} : (A \otimes -) \otimes C \Rightarrow A \otimes (- \otimes C)$ is a natural transformation for arbitrary objects A, C in $\mathcal{C}$,
- $a_{AB-} : (A \otimes B) \otimes - \Rightarrow A \otimes (B \otimes -)$ is a natural transformation for arbitrary objects A, B in $\mathcal{C}$,

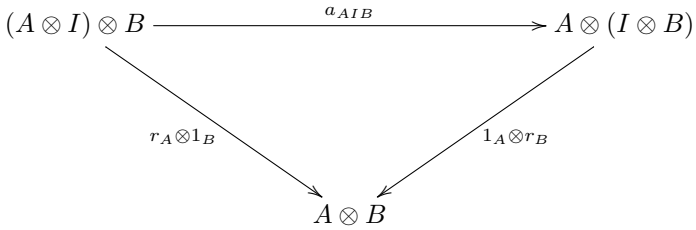
3. there is an object I in $\mathcal{C}$ called the *unit object* (or simply the *unit*),
4. for any object A in $\mathcal{C}$, there are isomorphisms:

$$l_A : I \otimes A \cong A \qquad \text{and} \qquad r_A : A \otimes I \cong A,$$

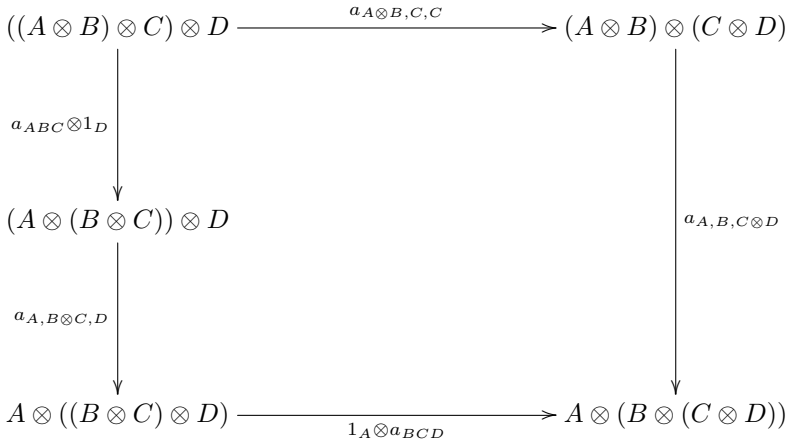
such that l_A and r_A are natural in A: both $l : I \otimes - \Rightarrow -$ and $r : - \otimes I \Rightarrow -$ are natural transformations

satisfying the following commutative diagrams:

- *unit coherence law*



- *associativity coherence law*



The bifunctor $\otimes$ is called the *tensor product* on $\mathcal{C}$, and the natural isomorphisms a, l, r are called the *associativity isomorphism*, the *left unit isomorphism*, and the *right unit isomorphism* respectively.

Some examples of monoidal categories are

- A prototype is the category of isomorphism classes of vector spaces over a field $\mathbb{K}$, herein the tensor product is the associative operation and the field $\mathbb{K}$ itself is the unit element.

- The category of sets is monoidal. The tensor product here is just the set-theoretic cartesian product, and any singleton can be used as the unit object.
- The category of (left) modules over a ring R is monoidal. The tensor product is the usual tensor product of modules, and R itself is the unit object.
- The category of bimodules over a ring R is monoidal. The tensor product and the unit object are the same as in the previous example.

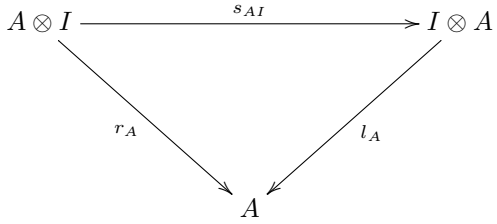
Symmetric monoidal categories

A monoidal category $\mathcal{C}$ with tensor product $\otimes$ is said to be *symmetric* if for every pair A, B of objects in $\mathcal{C}$, there is an isomorphism

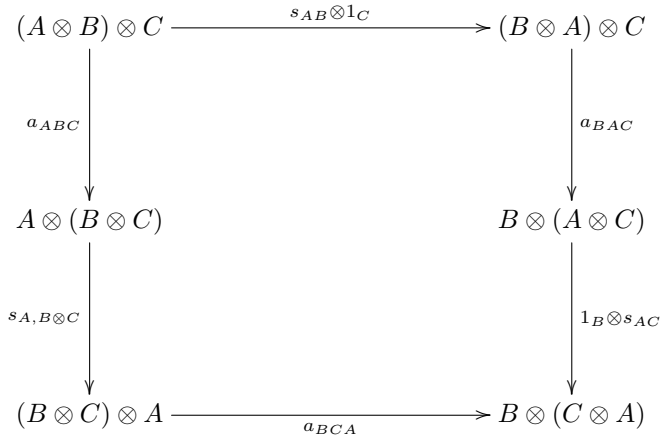
$$s_{AB} : A \otimes B \cong B \otimes A$$

that is natural in both A and B such that the following diagrams are commutative

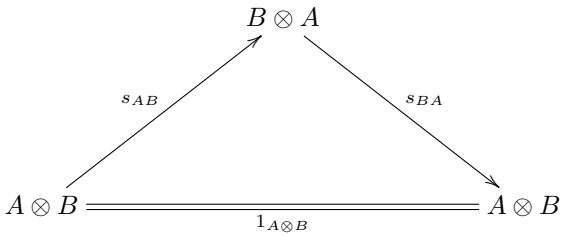
1. (*unit coherence for s*):



2. (*associativity coherence for s*):



3. (inverse law):



In the diagrams above, a, l, r are the associativity isomorphism, the left unit isomorphism, and the right unit isomorphism respectively.

Some examples and non-examples of symmetric monoidal categories:

- The category of sets. The tensor product is the set theoretic cartesian product, and any singleton can be fixed as the unit object.
- The category of groups. Like before, the tensor product is just the cartesian product of groups, and the trivial group is the unit object.
- More generally, a category with finite products is symmetric monoidal. The tensor product is the direct product of objects, and any terminal object (empty product) is the unit object.
- The category of bimodules over a ring R is monoidal. However, this category is only symmetric monoidal if R is commutative.

Remark. A symmetric monoidal category is a braided monoidal category such that the inverse law: $s_{BA} \circ s_{AB} = 1_{A \otimes B}$ holds.