Chapter 2

CONSTRUCTIONS

In this chapter we consider some fundamental categorical constructions, i. e., particular objects (and morphisms) that satisfy a given set of axioms described in the language of Category Theory. Since in this language there is no way to look at the internal membership structure of objects, all the concepts must be defined by their relations with other objects, and these relations are established by the existence and the equality of particular morphisms. This property of the categorical language, if compared to the traditional set-theoretic jargon, may be well understood by an analogy with computer science; namely, as we already mentioned, the categorical description corresponds to an abstract data specification, while the traditional set-theoretic approach is more similar to a concrete implementation.

2.1 Initial and Terminal Objects

2.1.1 Definition *Let* $\mathbf{C}$ *be a category. An object* 0 *is* ***initial*** *iff for any* $b \in Ob_{\mathbf{C}}$ *there is a unique* $f \in C[0,b]$.

The typical example of an initial object is the empty set $\emptyset$ in **Set**; indeed the empty function (i.e., the function whose graph is empty) is the unique arrow with $\emptyset$ for source.

A more interesting example is the following. Let $\mathbf{\Sigma}$ be a signature. The class $Alg_{\mathbf{\Sigma}}$ of $\mathbf{\Sigma}$-algebras with $\mathbf{\Sigma}$-homomorphisms as arrows forms a category. $Alg_{\mathbf{\Sigma}}$ has an initial object $T_{\mathbf{\Sigma}}$ which is called $\mathbf{\Sigma}$-word-algebras, or also Herbrand Universe for $\mathbf{\Sigma}$. The set $T_{\mathbf{\Sigma},s}$ (the carrier of $T_{\mathbf{\Sigma}}$ of sort s) is just the set of all well-formed expressions of sort s. If $\mathbf{\Sigma}$ is derived by a context free grammar (that is: sorts are nonterminals and operator symbols are productions of the grammar), then $T_{\mathbf{\Sigma},s}$ is the set of all parse trees for derivations in the grammar from the nonterminal s. In general the initial $\mathbf{\Sigma}$-algebra $T_{\mathbf{\Sigma}}$ corresponds to the syntax of a language of signature $\mathbf{\Sigma}$. Any other $\mathbf{\Sigma}$-algebras A in $Alg_{\mathbf{\Sigma}}$ is a possible semantic domain; the semantic function (interpretation) is the unique homomorphism from $T_{\mathbf{\Sigma}}$ to A.

Initiality is the simplest *universal* notion in Category Theory, since it is given by the existence and unicity of morphisms satisfying certain properties. This method is used everywhere in Category Theory.

2.1.2 Proposition *If* 0 *and* $0'$ *are two initial objects in a category* $\mathbf{C}$*, then they are isomorphic.*

Proof. Let $i: 0 \to 0'$, $j: 0' \to 0$ the morphisms respectively given by the initiality of 0 and $0'$. Then $j \circ i: 0 \to 0$, but also $id_0: 0 \to 0$, and since by initiality of 0, there is exactly one morphism in $\mathbf{C}[0,0]$, then $j \circ i = id_0$; in the same way, by initiality of $0'$ we have $i \circ j = id_{0'}$.♦

We will now show how *duality* can be used to define new concepts and to prove new assertions. Let P(c) be the property "for any $b \in Ob_{\mathbf{C}}$ there is a unique f, such that dom(f) = c, cod(f) = b." By definition c is initial iff P(c) holds; that is, P defines initiality. The dual statement of P is $P^{op}(c)$ = "for any $b \in Ob_{\mathbf{C}}$ there is a unique f, such that cod(f) = c, dom(f) = b." Usually the dual Q^{op} of a property Q defines a concept named by prefixing "co-" to the name of the property Q. In our case, we say that P^{op} defines coinitiality. An object c such that $P^{op}(c)$ holds, is called **co-initial**. Anyway it is common practice to assign to every coentity an independent name which better expresses its properties; for example, a coinitial object is known as **terminal** object. Note that an initial object is coterminal.

Terminal objects are usually represented with the number 1 or with the letter t. The unique morphism from an object a to the terminal object t is usually written $!a: a \to t$.

Any singleton set is terminal in **Set.** In the category **2** one object is initial and the other one is terminal. If c is initial in **C**, then it is terminal in the dual category $\mathbf{C}^{\mathbf{op}}$.

Consider now the statement P_1 = "*If 0 and 0' are two initial objects, then they are isomorphic*." Its dual is: P_1^{op} = "*If 0 and 0' are two terminal objects, then they are* isomorphic." (the property to be an isomorphism is the dual of itself: prove it as an exercise.) By our discussion of duality in chapter 1 and, since by proposition 2.1.2 P_1 holds in every category, P_1^{op} also does. We conclude the following:

2.1.3 Proposition *If 0 and 0' are two terminal objects in a category* ***C****, then they are isomorphic.*

Proof By duality and by proposition 2.1.2.♦

An object c in a category **C** may be both initial and terminal. An example is the unit group in **Grp**; in this case, it is called a **zero** object.

In **Set**, a morphism from the singleton {*} to a set A defines an element of A. For this reason an arrow from a terminal object t to an object a in a generic category **C** is usually called an **element** or a **point** of a. In this case, however, the set-theoretic intuition must be used very carefully, because it is quite common to work in categories where the categorical notion does not reflect the behavior of elements in **Set**. For example the set-theoretic intuition would suggest that every non-initial object must have at least one element: but consider the partial order category **3** which has three object $0 \leq 1 \leq 2$; clearly 0 is initial and 2 is terminal, 1 is non-initial but has no elements. Similarly, in **Set** two arrows are equal iff they coincide on all points, or, more formally, given functions f and g,

one has $f \neq g$ iff there is an element x of their domain such that $f \circ x \neq g \circ x$. However, in a generic category **C** with terminal object t, this is not necessarily true.

2.1.4 Definition. *Let* $\mathbf{C}$ *be a category.* $t \in Ob_{\mathbf{C}}$ *is a* ***generator*** *iff for all* $a,b \in Ob_{\mathbf{C}}$ *and all* $f,g \in \mathbf{C}[a,b]$, *one has:* $f \neq g \Rightarrow \exists h \in \mathbf{C}[t,a]\ f \circ h \neq g \circ h$.

$\mathbf{C}$ *has* ***enough points*** *(or is* ***well pointed****), if there exists a generator* t *that is terminal in the given category.*

In short, a category has enough points when the arrows from the terminal object allow to discriminate between morphisms, similarly as for elements over **Set**. Of course, it is not a surprise that the set-theoretic notions of "element" and of "extensionality" are somewhat awkward to deal with in the language of Category Theory.

2.2 Products and Coproducts

The categorical product is merely a "structural" generalization of the notion of Cartesian product of sets. Given two sets A and B, their cartesian product is:

$$A \times B = \{\langle x,y\rangle \,/\, x \in A,\ y \in B\}$$

Associated with this set there are two special maps $p_A: A \times B \to A$, $p_B: A \times B \to B$ called projections, such that for every $\langle x,y\rangle$ in $A \times B$ $p_A(\langle x,y\rangle) = x$, $p_B(\langle x,y\rangle) = y$. Note that for every c in $A \times B$, $\langle p_A(c), p_B(c)\rangle = c$.

Let C be another set, and $f: C \to A$, $g: C \to B$. Define $\langle f,g\rangle: C \to A \times B$ by $\langle f,g\rangle(c) = \langle f(c),g(c)\rangle$ for every $c \in C$. Then, for every $c \in C$, $p_A(\langle f,g\rangle(c)) = p_A(\langle f(c),g(c)\rangle) = f(c)$, that is, $p_A \circ \langle f,g\rangle = f$. In the same way, we obtain $p_B \circ \langle f,g\rangle = g$. Conversely, let $h: C \to A \times B$. Then for every $c \in C$, $\langle p_A \circ h, p_B \circ h\rangle(c) = \langle p_A(h(c)), p_B(h(c))\rangle = h(c)$, that is, $\langle p_A \circ h, p_B \circ h\rangle = h$.

The previous consideration suggests the following definition:

2.2.1 Definition *Let* $\mathbf{C}$ *be a category, and* $a,b \in Ob_{\mathbf{C}}$. *The* ***categorical product*** *of* a *and* b *is an object* $a \times b$ *together with two morphisms* $p_a: a \times b \to a$, $p_b: a \times b \to b$, *and for every object c an operation* $\langle\ ,\ \rangle_c : \mathbf{C}[c,a] \times \mathbf{C}[c,b] \to \mathbf{C}[c,a \times b]$ *such that for all morphisms* $f: c \to a$, $g: c \to b$, $h: c \to a \times b$, *the following equations hold:*

ia. $p_a \circ \langle f,g\rangle_c = f$;

ib. $p_b \circ \langle f,g\rangle_c = g$;

ii. $\langle p_a \circ h, p_b \circ h\rangle_c = h$.

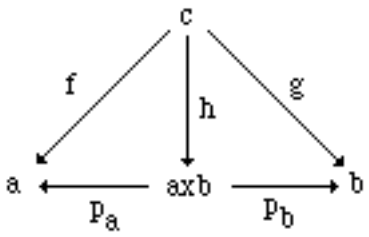
It is common practice to omit the subscript c in $\langle\ ,\ \rangle_c$ when its meaning is clear from the context.

The operation $< , > : \mathbf{C}[c,a]\times\mathbf{C}[c,b]\to\mathbf{C}[c,a\times b]$ of a categorical product is a bijection: its inverse is the operation that takes every arrow $h\in\mathbf{C}[c,a\times b]$ to the pair $(p_a \circ h, p_b \circ h)\in\mathbf{C}[c,a]\times\mathbf{C}[c,b]$.

The proof that these operations are inverse of each other is stated above in definition 2.2.1. Conversely, given a bijective operation $< , > : \mathbf{C}[c,a]\times\mathbf{C}[c,b]\to\mathbf{C}[c,a\times b]$ which satisfies (ia) and (ib), then (ii) is necessarily true. Indeed, let $h\in\mathbf{C}[c,a\times b]$. Then, since $< , >$ is bijective, there is a pair $(f,g)\in\mathbf{C}[c,a]\times\mathbf{C}[c,b]$ such that $h = <f,g>$; but $f = p_a \circ <f,g> = p_a \circ h$ and analogously, $g = p_b \circ <f,g> = p_b \circ h$; thus, $h = <p_a \circ h, p_b \circ h>$.

The last consideration leads us to a more compact but equivalent definition of a categorical product.

2.2.2 Definition *Let* $\mathbf{C}$ *be a category, and* $a,b\in Ob_{\mathbf{C}}$. *The* ***categorical product*** *of* a *and* b *is an object* $a\times b$ *together with two morphisms* $p_a: a\times b\to a$, $p_b: a\times b\to b$, *such that, for any* $f\in\mathbf{C}[c,a]$ *and* $g\in\mathbf{C}[c,b]$, *there exists exactly one* $h\in\mathbf{C}[c,a\times b]$ *such that the following diagram commutes*



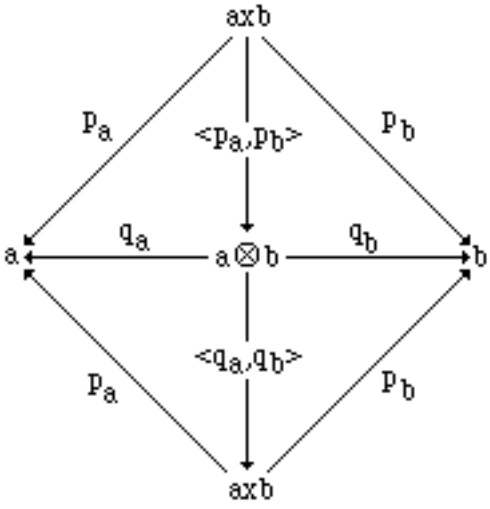
2.2.3 Definition *For* $f\in\mathbf{C}[a,c]$ *and* $g\in\mathbf{C}[b,d]$, *set* $f\times g = <f\circ p_a, g\circ p_b> : a\times b\to c\times d$.

Exercise Prove that for all arrows $h: e\to a$ and $k: e\to b$, $f\times g\circ<h,k> = <f\circ h, g\circ k>$.

2.2.4 Proposition *In a category, the product is unique (up to isomorphisms), if it exists.*

Proof Let $a\otimes b$ be an alternative product with projections q_a and q_b.

Then $<q_a,q_b> \circ <p_a,p_b>$ is the unique morphism such that the following diagram commutes:



Since $id_{a\times b}$ also does the same job, $id_{a\times b} = \langle p_a,p_b\rangle\circ\langle q_a,q_b\rangle$.
By symmetry, one also has $\langle p_a,p_b\rangle \circ \langle q_a,q_b\rangle = id_{a\otimes b}$.♦

Exercise. Prove the following facts:
1. $a \cong a'$ and $b \cong b'$ imply $a\times b \cong a'\times b'$.
2. $a\times b \cong b\times a$.

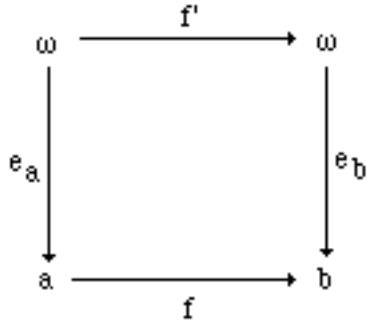
2.2.5 Definition *A category* ***C*** *is* ***Cartesian*** *(**C** is a CC) iff*

i. it contains a terminal object t;

ii. every pair $a,b\in Ob_{\mathbf{C}}$ *has a categorical product* $(a\times b, p_{a,b,1}: a\times b\to a, p_{a,b,2}: a\times b\to b)$

Exercises
1. Generalize the definition of a product of two objects to arbitrary products.
2. Prove that a Cartesian category **C** always contains all finite products.
3. Let **C** be a CC and let t be its terminal object. Prove that for all b in $Ob_{\mathbf{C}}$, $b \cong t\times b \cong b\times t$.

Examples The categories **Set**, **Top**, **Grp** are all Cartesian.
An interesting Cartesian category in Computability Theory is the category **EN** of numbered sets. Objects in **EN** are pairs $\underline{a} = (a,e_a)$, where a is a countable set and $e_a: \omega\to a$ is an onto map (an *enumeration* of a). $f\in\mathbf{EN}[\underline{a},\underline{b}]$ iff for some total recursive f' the following diagram commutes:



We say that f' **represents** f. The product is easily obtained by using any effective pairing of ω^2, $[,]: \omega\times\omega\to\omega$.

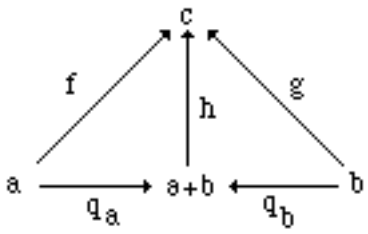
A typical numbered set which is worth studying is $\underline{PR} = (PR,\phi)$, the partial recursive functions with a Goedel numbering $\phi: \omega\to PR$. Then $\mathbf{EN}[\underline{PR},\underline{PR}]$ are exactly the type two recursive functionals. Of course, this is also a countable set. It is not trivial, though, to construct an "acceptable" enumeration of it. This will be an important issue in the sequel.

Exercises
1. Let $\underline{\omega} = (\omega,id)$ in **EN**. Then $f\in\mathbf{EN}[\underline{\omega},\underline{PR}]$ iff $\exists f'\in PR$ $f'([x,y]) = f(x)(y)$. Moreover, $g\in\mathbf{EN}[\underline{\omega},\underline{PR}]$ is principal iff g is an acceptable Goedel numbering of PR, in the sense of classical recursion theory.

2. Let **C** be a CC, and V be an object such that $V \times V < V$. Then the category of retractions on V (see definition 1.4.4) is a CC.

The dual of the notion of a product is the **coproduct** a+b with embeddings q_1, q_2.

2.2.6 Definition. *Let* ***C*** *be a category, and* $a,b \in Ob_{\mathbf{C}}$. *The* ***coproduct*** *of a and b is an object a+b together with two morphisms* $q_a: a \to a+b$, $q_b: b \to a+b$ *such that, for any* $f \in \mathbf{C}[a,c]$ *and* $g \in \mathbf{C}[b,c]$, *there exists exactly one* $h \in \mathbf{C}[a+b,c]$ *such that the following diagram commutes*



By duality, the coproduct is unique (up to isomorphisms).

Examples

1. In **Set** the coproduct is the disjoint union.
2. In a preorder **P** the product is the greatest lower bound, if it exists. The coproduct is the least upper bound, if it exists.
3. Let **CPO** be the category of complete partial orders with continuous functions with respect to the order or Scott topology. **CPOS** is the subcategory with only strict functions, i.e., morphisms always take the least element $\perp$ to the least element of the target space. It is easy to see that both categories are Cartesian. The coproduct in **CPOS** is given by the coalesced sum, i.e., the disjoint union except for the identification of the two least elements. On the other hand, there is no coproduct in **CPO**. This may be seen by observing that in **CPO** one may have $f(\perp) \neq g(\perp)$, by which the coalesced sum fails to give a coproduct; an extra common least element (disjoint sum) may give more than one extension of the required $\langle f,g \rangle^{op}$.

2.3 Exponentials

In the connection we mentioned between Category Theory and Computation Theory, as "theories of functions," a fundamental aspect still has to be taken care of. In either case, we may be interested in computing with procedures as arguments. That is, we may need to describe higher type functions.

So far we have only become familiar with Cartesian categories, where the object a×b, representing the product, is defined. Thus, the notion of morphism taking morphisms as arguments doesn't yet make sense. What we first need, then, is a further closure property, namely, the existence within the

category of an object b^a which suitably represents the set of morphisms from b to a. With an informal reference to typing in programming, the key property of the objects, which represent the sets of morphisms, provides an interpretation to a common construct in actual programming, namely, the identification of types such as $A\times B\to C$ and $A\to(B\to C)$. This corresponds to the following important uniformity property of programs of several arguments, which is directly inherited from classical Recursion Theory.

Let $\{\phi_i\}_{i\in\omega} = PR$ be an acceptable Gödel numbering of the partial recursive functions and $[,]: \omega\times\omega\to\omega$ be an effective pairing. Define then, as usual, $f: \omega\times\omega\to\omega$ is a binary partial recursive function iff $\exists f'\in PR\ f(x,y) = f'([x,y])$ (similarly, for n-ary functions, $n \geq 2$). By this and by the s-m-n iteration theorem one immediately has $f: \omega\times\omega\to\omega$ is partial recursive iff $\exists s\in R\ \phi_{s(x)}(y) = f(x,y)$.

Thus, a two-(or more) argument function f is computable iff it is computable in each argument and the function $x \mapsto f(x,_)$ is also "computable," i.e., $\exists s\in R\ \phi_{s(x)} = f(x,_)$. In other words, in computability theory, f is in $\omega\times\omega\to\omega$ iff $x \mapsto f(x,_)$ is in $\omega\to(\omega\to\omega)$. Similarly, the category-theoretic closure property we need concerns the existence, for any $f: c\times a\to b$, of a morphism within the category, which does the same job as s or $x \mapsto f(x,_)$ in recursion theory. We will call it $\Lambda(f)$.

Exercise For $n \geq 2$, not every n-ary function which is computable in each argument needs to be computable. (Hint: take g total nonrecursive and set $f(x,y) = g(\min\{x,y\})$).

2.3.1 Definition *Let* $\mathbf{C}$ *be a Cartesian category, and* $a,b\in Ob_{\mathbf{C}}$. *The* ***exponent*** *of a and b is an object* b^a *together with a morphism* $eval_{a,b}: b^a\times a\to b$ *(evaluation map), and for every object c an operation* $\Lambda_c : \mathbf{C}[c\times a,b]\to\mathbf{C}[c,b^a]$ *such that for all morphisms* $f: c\times a\to b$, $h: c\to b^a$, *the following equations hold:*

$$\beta).\quad eval_{a,b}\circ(\Lambda(f)\times id_a) = f\ ;$$

$$\eta).\quad \Lambda_c(eval_{a,b}\circ(h\times id_a)) = h\ .$$

(We may omit the indices when unambiguous, as usual.)

In **Set** the exponent set of A and B is $B^A = \{f \mid f \text{ is a function from } A \text{ to } B\}$, thus $B^A = \mathbf{Set}[A,B]$. The function $eval: B^A\times A\to B$ is given by the rule: $eval(\langle f,x\rangle) = f(x)$.

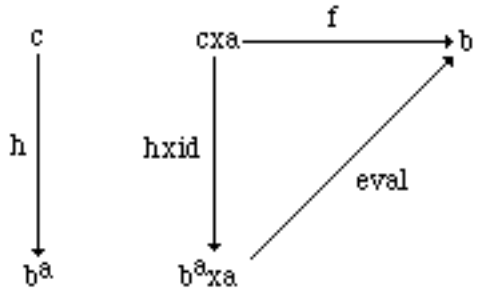
$\Lambda: \mathbf{Set}[C\times A,B]\to\mathbf{Set}[C,B^A]$ takes every function $f: C\times A\to B$ to the function $\Lambda(f): C\to B^A$ defined by $\Lambda(f)(c) = \lambda a.f(c,a)$, where $\lambda a.f(c,a)\in B^A=\mathbf{Set}[A,B]$ is the function which takes $a\in A$ to $f(c,a)\in B$. The proof of (β) and (η) is almost immediate.

As in the case of the product, observe that in general the operation $\Lambda: \mathbf{C}[c\times a,b]\to\mathbf{C}[c,b^a]$ in definition 2.3.1 is a bijection. Indeed, by (β) and (η), Λ^{-1} is the operation which takes every $h\in\mathbf{C}[c,b^a]$ to $eval_{a,b}\circ(h\times id_a)\in\mathbf{C}[c\times a,b]$.

Conversely, if $\Lambda : \mathbf{C}[c\times a,b]\to\mathbf{C}[c,b^a]$ is a bijection and (β) holds, then (η) is necessarily true. Indeed, let $h\in\mathbf{C}[c,b^a]$ and take $f\in\mathbf{C}[c\times a,b]$ such that $h = \Lambda(f)$; then $\Lambda(eval_{a,b}\circ(h\times id_a)) = \Lambda(eval_{a,b}\circ(\Lambda(f)\times id_a)) = \Lambda(f) = h$.

The following is thus an equivalent definition of "exponent":

2.3.2 Definition *Let* ***C*** *be a Cartesian category and* $a,b\in Ob_{\mathbf{C}}$. *The* ***exponent*** *of* a *and* b *is an object* b^a *together with a morphism* $eval_{a,b}: b^a\times a\to b$, *such that for all morphisms* $f: c\times a\to b$, *there exists one and only one* $h: c\to b^a$ *such that the following diagram commutes:*



Exercise By setting $\Lambda(f) = h$, give the details of the equivalence proof between the two definitions.

The previous diagram should suggest in which sense b^a "represents" $\mathbf{C}[a,b]$. The eval morphism generalizes the set-theoretic evaluation function $eval(f,x) = f(x)$. Moreover, take $c = t$, the terminal object. Then $\mathbf{C}[t,b^a] \cong \mathbf{C}[t\times a,b] \cong \mathbf{C}[a,b]$ as sets. This is particularly significant if **C** has enough points (why?).

2.3.3 Definition. ***C*** *is a* ***Cartesian closed category*** *(CCC) iff*

1. ***C*** *is cartesian,*

2. for every pair $a,b\in Ob_{\mathbf{C}}$, *there is an exponent.*

Set is a CCC: the previous definition of exponents in **Set** clearly holds for every pair of sets. Another simple CCC is **CPO**, the category of complete partial orders and continuous maps. As well-known, given c.p.o.'s a and b, $\mathbf{CPO}[a,b]$ is also a c.p.o., with respect to the pointwise ordering. Moreover, both $eval$ and $\Lambda(f)$, defined as for **Set** by using continuous functions, are continuous and satisfy the required conditions. Note that the proof uses the well-known fact that in **CPO** a function is continuous iff it is so in each argument and the map $x \mapsto \lambda y.f(x,y)$ is continuous. Actually, even Λ is continuous.

Among the various examples of categories mentioned in these notes, an important one does not satisfy Cartesian closedness: the category **EN** in section.2.2. Consider, say, $\underline{\omega} = (\omega,id)$. Then $\mathbf{EN}[\underline{\omega},\underline{\omega}]$ (= R, the recursive functions) is surely countable. However, if a numbered set (ω^ω,φ) and

a morphism eval with the above properties existed, then $u(x,y) = eval(\varphi(x),y)$ would be a universal function for R.

Also, the ω-algebraic c.p.o.'s, that is, the c.p.o.'s with a countable collection of compact elements approximating all the others (see Scott domains below) and continuous maps as morphisms do not form a CCC. They contain, though, some fundamental subCCC's for the purposes of denotational semantics of programming languages and higher type Recursion Theory. They will be explored in the examples below.

Given a CCC **D**, it may be interesting to consider specific "structures of types" in it. That is, for a collection A of objects in **D**, let $\mathbf{D}_A$ be the full **sub CCC generated by** A in **D**, i.e., the least full sub category such that $A \subseteq \mathbf{D}_A$ and $a,b \in \mathbf{D}_A \Rightarrow a\times b, a^b \in \mathbf{D}_A$.

Exercise Prove that in any CCC one has $a^{b\times c} \cong (a^b)^c$.

In definition 1.4.2, we introduced the notion of "retract": in a category **C**, $a<b$ via the retraction (i,j) iff $j \circ i = id_a$. In these assumptions, i turns out to be mono and j epic. Thus, a retract a of b is a subobject of b in the sense of section 1.5. In the case of **Set**, nonempty subsets and retracts happen to coincide, as surjections from a set to a subset are always possible. In more structured categories this reinforcement of the idea of subset, given by retractions, turns out to be very informative. In particular, we will discuss categories with nontrivial objects a such that $a^a<a$. This is clearly impossible in **Set** because, by Cantor's theorem, the cardinality of the exponent a^a, when a is not a singleton, is strictly bigger than the cardinality of a. In short, we will put together retracts and exponents, in a nontrivial way, in order to discuss one of the early relevant applications of categorical notions to computer science, namely the invention of mathematical (categorical, to be precise) models of type-free languages. In these languages, programs are viewed as data or, semantically, exponents may be retracted into (source and target) objects. It is convenient to prove, in general, some basic properties of exponents and retractions for their relevance and simplicity as well as for some preliminary training on equational reasononing, which will turn out to be useful to the reader in the sequel.

2.3.4 Proposition *Let **C** be a CCC. If $a<a'$ (via $in_a: a\to a'$, $out_a: a'\to a$), and $b<b'$ (via $in_b: b\to b'$, $out_b: b'\to b$), then $b^a<b'^{a'}$, via $\Lambda(in_b\circ eval\circ(id\times out_a)): b^a\to b'^{a'}$, $\Lambda(out_b\circ eval\circ(id\times in_a)): b'^{a'}\to b^a$.*

Proof. $\Lambda(out_b\circ eval\circ(id\times in_a)) \circ \Lambda(in_b\circ eval\circ(id\times out_a)) =$

$= \Lambda(out_b\circ eval\circ(id\times in_a)\circ\Lambda(in_b\circ eval\circ(id\times out_a))\times id)$

$= \Lambda(out_b\circ eval\circ\Lambda(in_b\circ eval\circ(id\times out_a))\times id\circ(id\times in_a))$

$= \Lambda(out_b\circ(in_b\circ eval\circ id\times out_a)\circ(id\times in_a))$

$= \Lambda(eval \circ id\times(out_a\circ in_a))$

$= \Lambda(\text{eval} \circ \text{id}\times\text{id})$
$= \text{id}.$ ♦

2.3.5 Definition *Let* ***C*** *be a CCC. An object* V *of* ***C*** *is* ***reflexive*** *iff* $V^V < V$.

Before we get to see some reflexive objects in relevant CCC's in the following sections, it is worth proving two simple, but general, properties of reflexive objects (see chap.8 for applications).

2.3.6 Proposition *Let* ***C*** *be a CCC, and* V *a reflexive object. Then* $t < V$ *and* $V\times V < V$.
Proof. Let ($\mathbf{in}$: $V^V \to V$, $\mathbf{out}$: $V \to V^V$) the retraction pair between V^V and V. In order to prove that $t<V$ we must only prove the existence of a morphism from t to V (why?). Let then p_1: $t\times V \to V$ be the projection; $\Lambda(p_1)$: $t \to V^V$, and thus $\mathbf{in} \circ \Lambda(p_1)$: $t \to V$.
The proof that $V\times V<V$ is much more complex; we prove that $V\times V<V^V$; then $V\times V<V$ follows by composition.

Let app = eval$\circ$($\mathbf{out}\times \text{id}_V$): $V\times V \to V$, and let $\alpha_{a,b,c}$ be the isomorphism $\alpha_{a,b,c}$: $(b\times c)\times a \to (a\times b)\times c$. Then: app $\circ$ (app$\times$id) $\circ$ $\alpha_{V,V,V}$: $(V\times V)\times V \to V$
and $\mathbf{in_1} = \Lambda(\text{app} \circ (\text{app}\times\text{id}) \circ \alpha) : (V\times V) \to V^V$.
By proposition 2.3.4 one has $(V^V)^V < V^V$ via

$\mathbf{in_2} = \Lambda(\mathbf{in} \circ \text{eval} \circ (\text{id}\times\text{id})) : (V^V)^V \to V^V$
$\mathbf{out_2} = \Lambda(\mathbf{out} \circ \text{eval} \circ (\text{id}\times\text{id})) : V^V \to (V^V)^V$.

Let p_2: $t\times V \to V$ and pr_1:$V\times V \to V$, pr_2: $V\times V \to V$ be the projections respectively associated with the products $t\times V$ and $V\times V$. Then, for i =1,2, $\Lambda(pr_i)$: $V \to V^V$ and, thus, for $\Lambda(\Lambda(pr_i) \circ p_2)$: $t \to (V^V)^V$, $\mathbf{p_i} = \mathbf{in} \circ \mathbf{in_2} \circ \Lambda(\Lambda(pr_i) \circ p_2)$: $t \to V$.

Define, then, $\mathbf{out_1} = \langle \text{eval} \circ \langle \text{id}, \mathbf{p_1} \circ !V^V \rangle, \text{eval} \circ \langle \text{id}, \mathbf{p_2} \circ !V^V \rangle \rangle : V^V \to V\times V$.

We must prove that $\mathbf{out_1} \circ \mathbf{in_1} = \text{id}_{V\times V}$, or equivalently that for i = 1,2, $pr_i \circ \mathbf{out_1} \circ \mathbf{in_1} = pr_i$.

$pr_i \circ \mathbf{out_1} \circ \mathbf{in_1} = \text{eval} \circ \langle \text{id}, \mathbf{p_i} \circ !V^V \rangle \circ \mathbf{in_1}$
$= \text{eval} \circ \langle \Lambda(\text{app} \circ (\text{app}\times\text{id}) \circ \alpha), \mathbf{p_i} \circ !V\times V \rangle$
$= \text{app} \circ (\text{app}\times\text{id}) \circ \alpha \circ \langle \text{id}_{V\times V}, \mathbf{p_i} \circ !V\times V \rangle$
$= \text{app} \circ \langle \text{app} \circ \langle \mathbf{p_i} \circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \text{eval}\circ(\mathbf{out}\times\text{id}_V) \circ \langle \mathbf{in}\circ\mathbf{in_2}\circ\Lambda(\Lambda(pr_i) \circ p_2) \circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \text{eval}\circ\langle \mathbf{in_2}\circ\Lambda(\Lambda(pr_i) \circ p_2) \circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \text{eval}\circ\langle \Lambda(\mathbf{in}\circ\text{eval}\circ(\text{id}\times\text{id}))\circ\Lambda(\Lambda(pr_i) \circ p_2) \circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \text{eval}\circ\langle \Lambda(\mathbf{in}\circ\text{eval}\circ(\Lambda(\Lambda(pr_i) \circ p_2)\times\text{id}))\circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \text{eval}\circ\langle \Lambda(\mathbf{in} \circ \Lambda(pr_i) \circ p_2)\circ !V\times V, pr_1 \rangle, pr_2 \rangle$
$= \text{app} \circ \langle \mathbf{in} \circ \Lambda(pr_i) \circ pr_1, pr_2 \rangle$
$= \text{eval}\circ(\mathbf{out}\times\text{id}_V) \circ (\mathbf{in} \circ \Lambda(pr_i))\times\text{id}_V$

$= \text{eval} \circ (\Lambda(pr_i) \times id_V)$

$= pr_i$. ♦

Exercises For the following exercises, assume that **C** is a CCC.

1. Let V be a reflexive object of **C**. Prove that the collection $\mathbf{Ret_V}$ of all retracts of V in **C** is a CCC.

2. (Difficult, see section 8.8) Let b be an object of **C.** A **fixpoint operator** for b is a morphism $Fix_b: b^b \to b$ such that $Fix_b = eval_{b,b} \circ \langle id, Fix_b \rangle$. Let $V^V < V$ via (in,out). Let also

$F = eval \circ \langle id, in \rangle : V^V \to V$;

$H = \Lambda(eval \circ (id \times (F \circ out))) : V^V \to V^V$.

Prove that $F \circ H$ is a fixpoint operator for V. Define a fixpoint operator for all objects in $\mathbf{Ret_V}$.

3. Let **C** be a CCC and suppose that for all a, b in $Ob_{\mathbf{C}}$ there exists the coproduct a+b (with embedding $in_a: a \to a+b$, $in_b: b \to a+b$). Prove that, for all c in $Ob_{\mathbf{C}}$, $(a \times c)+(b \times c)$ is isomorphic to $(a+b) \times c$, and define explicitly the isomorphism.

Result: $(a \times c)+(b \times c) \cong (a+b) \times c$ via

$i_1 = (in_a \times id_c)+(in_a \times id_c): (a \times c)+(b \times c) \to (a+b) \times c$

$i_2 = \Lambda^{-1}(\Lambda(in_{a \times c})+\Lambda(in_{b \times c})): (a+b) \times c \to (a \times c)+(b \times c)$

Proving $i_2 \circ i_1 = id$ is easy. For $i_1 \circ i_2 = id$, note first that $g \circ \Lambda^{-1}(f) = \Lambda^{-1}(\Lambda(g \circ eval) \circ f)$.

Then, in a few steps, one obtains

$i_1 \circ i_2 = \Lambda^{-1}(\Lambda(in_a \times id_c)+\Lambda(in_b \times id_c))$

$= \Lambda^{-1}(\Lambda(id) \circ in_a + \Lambda(id) \circ in_b)$

$= \Lambda^{-1}(\Lambda(id))$

$= id$

2.4 Examples of CCC's

2.4.1 Scott Domains

In this section and in the following one we introduced two fundamental examples of CCC's, namely, Scott domains and coherent domains. We define only the exponent object and the eval function and check that they are respectively an object and an arrow of the category. We leave the problem of defining the isomorphism Λ and checking (βcat) and (ηcat) as an (easy) exercise for the reader.

2.4.1.1 Definition *Let $(X,\leq)$ be a partiallly ordered set (po-set).*

*$D \subseteq X$ is **directed** iff it is nonempty and, for any $i,j \in D$, there is $k \in D$ such that $i \leq k$, $j \leq k$. A p.o.set $(X,\leq)$ is **complete** (is a **CPO**) iff every directed subset $D \subseteq X$ has a least upper bound $\cup D$ (the least element $\perp$ is the least upper bound of the empty directed set).*

*A point $x \in X$ is **compact** (**finite**) if for every directed D such that $x \leq \cup D$, there is an element $y \in D$ such that $x \leq y$. Let X_0 denote the collection of compact elements of X.*
*The c.p.o. $(X, \leq)$ is **algebraic** if for every $x \in X$ the set $x\downarrow = \{x_0 \in X_0 \mid x_0 \leq x\}$ is directed and $\cup(x\downarrow) = x$.*
*A c.p.o. $(X, \leq)$ is **bounded complete** if every bounded subset of X has a least upper bound. A **Scott Domain** is a bounded complete algebraic c.p.o..*

Exercises

1. Check that $\{\{y \mid x_0 \leq y\} \mid x_0 \in X_0\}$ is a basis for a T_0 topology on a Scott Domain. This topology is usually called **Scott topology**.
2. Prove that the least upper bound of a finite set of finite elements is always finite, if it exists.
3. (Nontrivial) Find counterexamples for the following assertions:
   i. if x_0 is compact then the set $\{y \mid y \leq x_0\}$ is finite;
   ii. if x_0 is compact and $y \leq x_0$ then y is compact.

2.4.1.2 Definition *Let $(X, \leq_X)$, $(Y, \leq_Y)$ be c.p.o.'s. A function $f: X \to Y$ is **monotonic** if it is order preserving, i.e., $i \leq_X j$ implies $f(i) \leq_Y f(j)$. (We will often omit the subscript X in $\leq_X$.)*
*A function $f: X \to Y$ is **continuous** if for every directed $D \subseteq X$, $f(\cup D) = \cup_{d \in D} f(d)$.*

Exercise Let $(X, \leq_X)$, $(Y, \leq_Y)$ be Scott domains. Prove that a function $f: X \to Y$ is continuous according to the previous definition iff it is continuos with respect to the Scott topology.

2.4.1.3 Definition *The category **D** has Scott domains for objects and continuous functions for morphisms. Let X, Y be objects of **D**. Y^X is just the collection of the continuous functions from X to Y ordered pointwise.*

Of course Y^X is a c.p.o.. We have to prove that it is bounded complete and algebraic.
In order to show that Y^X is bounded complete, assume that $\{f_i\}_{i \in I}$ has an upper bound g. Define then h by $h(x) = \cup_{i \in I}\{f_i(x)\}$. The function h is well defined since the set $\{f_i(x)\}_{i \in I}$ is bounded by g(x) and, thus, it has a least upper bound in Y. Moreover, h is continuous because for every directed set D in X one has:

$$\begin{aligned} h(\cup D) &= \cup_{i \in I}\{f_i(\cup D)\} \\ &= \cup_{i \in I} \cup_{x \in D} \{f_i(x)\} && \text{by the continuity of } f_i \\ &= \cup_{x \in D} \cup_{i \in I} \{f_i(x)\} \\ &= \cup_{x \in D} \{h(x)\} \end{aligned}$$

It easy to check that h is a least upper bound for $\{f_i\}_{i \in I}$.
To show that Y^X is algebraic, we explicitly define the set $(Y^X)_0$ of its compact elements.

2.4.1.4 Definition *A **step funcion** from X to Y is a function step-a,b where $a\in X_0$, $b\in Y_0$, defined by: step-a,b (x) = if $a \le x$ then b else $\perp$.*

We claim that the compact element of Y^X are exactly the least upper bound of finite bounded sets of step functions. In other words, for every $f_0\in(Y^X)_0$, (*) $f_0 = \cup_{i\in I}\{\text{step-}a_i,b_i\}$ for some finite I.

Let us prove first that every function $f = \cup_{i\in I}\{\text{step-}a_i,b_i\}$ is compact, when I is finite and $\cup_{i\in I}\{\text{step-}a_i,b_i\}$ exists, i.e., when for all subset J of I, $a_J = \cup_{i\in J}\{a_i\}$ exists $\Rightarrow$ $b_J = \cup_{i\in J}\{b_i\}$ exists (We then say that I is a **compatible** set of indices.)

Then, let $\{g_h\}_{h\in D}$ be a directed family in Y^X such that $f \le \cup_{h\in D}\{g_h\}$. In particular, for every J in I as above, $f(a_J) = b_J \le (\cup_{h\in D}\{g_h\})(a_J) = \cup_{h\in D}\{g_h(a_J)\}$. Clearly, for each J, b_J is compact and $\{g_h(a_J)\}_{h\in D}$ is directed. Let then $b_J \le g_{h(J)}(a_J)$ for some $h(J)\in D$. Since I is finite and $\{g_h\}_{h\in D}$ is directed, let g_k, for $k\in D$, be such that $g_k \ge g_{h(J)}$ for all J in I. Clearly $f \le g_k$ and we are done.

Prove now for exercise that for every continuous function $f: X\to Y$, one has

i. the set $F = \{ \cup_{i\in I}\{\text{step-}a_i,b_i\} \mid$ I finite, and $b_i \le f(a_i)\}$ is directed

ii. $f = \cup F$

Suppose then that f is compact. We need to prove that $f = \cup_{i\in I}\{\text{step-}a_i,b_i\}$ for some finite compatible I. By the exercise, $f = \cup F$, for F directed; thus, there exists I such that $\cup_{i\in I}\{\text{step-}a_i,b_i\}\in F$ and $\cup_{i\in I}\{\text{step-}a_i,b_i\} \ge f = \cup F \ge \cup_{i\in I}\{\text{step-}a_i,b_i\}$. In conclusion $f = \cup_{i\in I}\{\text{step-}a_i,b_i\}$; that is, every finite element in X^Y has the form (*) and, in particular, X^Y is a Scott domain.

The function $\text{eval}_{X,Y}: Y^X\times X\to Y$ is defined by $\text{eval}_{X,Y}(f,x) = f(x)$. The proof that eval is continuous is straightforward.

Interesting examples of Scott Domains may be found everywhere in the literature of denotational semantics. Indeed, the Cartesian closedness of the category allows you to construct plenty of them as products and exponents over commonly used ground types. That is, consider your preferred types of data (integers, booleans, strings,etc.). Organize them as flat p.o.sets, i.e., add a least element $\perp$ and set $x \le x'$ iff $x = \perp$ or $x = x'$. These are clearly objects of **D** as well as their products and exponents.

Other relevant examples are given, for example, by the p.o.sets P of the partial maps from ω to ω, the natural numbers, and Pω, the powerset of ω. The partial order, in these cases, is given by set inclusion, which on P means graph inclusion of functions, i.e., $f \le g$ iff $\forall n$ $(f(n)\downarrow \Rightarrow g(n) = f(n))$. As an exercise, the reader may check that both P and Pω live in **D**.

Interestingly enough, these two familar structures are also reflexive objects in **D** . We sketch the proof of this for Pω , see section 9.6-2 for more on P.

Let $\{e_n\}_{n\in\omega}$ be a canonical (bijective and effective) enumeration of the finite subsets of ω and let $<,>: \omega\times\omega\to\omega$ be a canonical coding of pairs. Define then graph: $\mathbf{D}[P\omega,P\omega]\to P\omega$ by graph(f) = $\{<n,m> \mid m\in f(e_n)\}$ and fun: $P\omega\to\mathbf{D}[P\omega,P\omega]$ by fun(a)(b) = $\{m \mid \exists e_n \subseteq b\ <n,m>\in a\}$. It is a simple exercise to check that graph and fun are morphisms in **D**. Moreover, fun $\circ$ graph = id and, thus, $P\omega^{P\omega}<P\omega$.

This example, which played a relevant role in denotational semantics, has been directly inspired by Recursion Theory (see the references). Indeed, the work carried on so far can be naturally "effectivized."

2.4.1.5 Definition *A Scott domain* $\underline{X} = (X,\leq)$ *is* ***effectively given*** *if* $\exists e_0: \omega\to X_0$ *bijective and*

1. $\exists z\in X\ e_0(n), e_0(m) \leq z$ *is decidable in* n, m
2. $\exists g\in R\ (\exists z\in X\ e_0(n), e_0(m) \leq z \Rightarrow e_0(g(n,m)) = sup\{e_0(n),e_0(m)\})$.

Call **ED** the category of effectively given Scott domains and continuous functions. **ED** is a CCC. As a matter of fact, the effectiveness properties are easily inherited at higher types.

Observe that, instead of taking the least upper bounds (l.u.b.'s) of all directed sets, as required in the definition of **ED**, one may take only the **computable** l.u.b.'s, i.e. the l.u.b.'s of directed sets or ideals in (X_0,e_0) that are indexed over recursively enumerable (r.e.) sets. (One may independently choose directed sets or ideals and obtain the same collection of computable elements.)

These limits are computable in a very sound sense. For example, $(P\omega, \{e_n\}_{n\in\omega}, \subseteq)$ is in **ED** and its computable elements are exactly the r.e. sets.

Exercise Prove a similar fact for the set P of partial maps from ω to ω.

Call **constructive domain** a domain whose elements are the computable elements in an effectively given domain. Since **ED** is Cartesian closed, this may be done in any (higher) type. In particular, given the constructive domains $\underline{X}_c, \underline{Y}_c$ obtained from $\underline{X}$ and $\underline{Y}$, one may consider the constructive domain $\underline{Y}^{\underline{X}}_c$ of the computable elements of $\underline{Y}^{\underline{X}}$. Define then the following

2.4.1.6 ***CD*** *is the category of constructive domains and continuous and computable morphisms.*

Exercise One clearly has to check that, for $f\in\underline{Y}^{\underline{X}}_c$, $\forall x\in X_c$ $f(x)\in Y_c$.

By the Cartesian closedness of **ED**, **CD** also is a CCC. Observe that each $\underline{X}_c$ is countable and that it can be effectively enumerated by using an acceptable enumeration of the r.e. sets. Typical objects in **CD** are RE, the recursively enumerable sets, and PR (= P_c), the partial recursive functions (see the exercise above).

Thus, in a rather indirect way, that is by topological and order properties, we obtained a CCC of countable (and numbered) sets. The proof that **CD** is a full sub-CCC of **EN** requires an important generalization, in higher types, of the classical Myhill-Shepherdson theorem for enumeration operators. The main application of **CD** is the characterization of the partial (continuous) and computable functionals as the sub-CCC of **CD** generated by PR, i.e., taking PR and constructing all higher types within **CD** (see also section 8.4-I). Moreover, one can give a countable and effective interpretation to the recursive definitions of programs and data types within **CD** (by a constructive version of the "limit constructions" in chapter 10).

Exercise Prove that RE is reflexive in **CD** (use the full and faithful embedding of **CD** in **EN**).

2.4.2 Coherent Domains

2.4.2.1 Definition *A **coherent structure** is a pair $(|X|, \uparrow)$, where $|X|$ is a set and $\uparrow$ is a binary, reflexive, symmetric relation on $|X|$. The elements of $|X|$ are called **points**, and the relation $\uparrow$ is called **coherence**.*
*The **coherent domain** associated with $(|X|, \uparrow)$ is the collection X of subsets of $P(|X|)$ whose points are pairwise coherent. The elements of X are ordered by set-inclusion.*
Coherence is extended to X in the obvious way, that is: $A \uparrow B$ iff $A \cup B \in X$.

Exercise Prove, when X is a coherent domain, that
1. $\emptyset \in X$
2. X is closed under directed union
3. ($A \in X$ and $B \subseteq A$) $\Rightarrow$ $B \in X$

2.4.2.2 Definition *Let X, Y be two coherent domains. A function $F: X \to Y$ is **stable** iff*
i). F is continuous
ii. $\forall A, B \in X \quad A \uparrow B \Rightarrow F(A \cap B) = F(A) \cap F(B)$

2.4.2.3 Definition *The category **Stab** has coherent domains as objects and stable functions as morphisms.*

Given two coherent domains X and Y, their **product** $X \times Y$ is defined by:
i $|X \times Y| = \{(0,z) / z \in |X|\} \cup \{(1,z) / z \in |Y|\}$
ii. $(a,z) \uparrow (a',z') \ [\text{mod } X \times Y]$ iff $a = a' \Rightarrow z \uparrow z' \ [\text{mod } D(a)]$, where $D(0) = X$ and $D(1) = Y$.

Exercise Define the projections and check that they are stable, i.e., prove that **Stab** is Cartesian.

There is simple way to obtain stable functions over coherent domains.

2.4.2.4 Definition *Let X,Y be coherent domains. Let also f be an injective function from $|X|$ to $|Y|$ such that, for all $x,x' \in |X|$, one has $\{x, x'\} \in X \Leftrightarrow \{f(x), f(x')\} \in Y$. Define then $f^+ : X \to Y$ and $f^- : Y \to X$ by*

i. $f^+(a) = \{f(z) / z \in a\}$

ii. $f^-(b) = \{z / f(z) \in b\}$

It is a matter of a simple exercise to prove that both f^+ and f^- are stable functions.

We need to construct next an exponent object out of the set of stable maps over coherent domains.

2.4.2.5 Definition *Let $F: X \to Y$ be a stable function. The **Trace** of F is $Tr(F) = \{(a,z) / a \in X,$ a is finite, $z \in |Y|$, $z \in F(a)$, $(\forall a' \subseteq a , z \in F(a') \Rightarrow a = a')\}$.*

F is completely determined by its trace by means of the following equation: $F(A) = \{z \in |Y| / \exists a \subseteq A$ $(a,z) \in Tr(F)\}$.

Exercise Prove that the correspondence between stable functions and their traces is bijective.

Notation The symbol $\uparrow\uparrow$ is used to represent **strict coherence**, i.e., $A \uparrow\uparrow B$ iff $A \uparrow B$ and $A \neq B$.

2.4.2.6 Definition *Let $|Y^X| = \{(a,z) / a \in X, a$ is finite, $z \in |Y| \}$. Moreover, let $(a,z) \uparrow (a',z')$ iff*

i. $a \uparrow\uparrow a' \ [mod\ X] \Rightarrow z \uparrow\uparrow z' \ [mod\ Y]$ *and*

ii. $a \uparrow a' \ [mod\ X] \Rightarrow z \uparrow z' \ [mod\ Y]$.

*Then Y^X is the **arrow domain** (exponent object).*

Exercises

1. Prove that conditions (i) and (ii) may be stated equivalently as

   $(a,z) = (a',z')$ or $z \uparrow\uparrow z'$ or **not** $a \uparrow a'$.

2. Prove that every element of Y^X is a trace of some stable function from X to Y, and conversely that if $F: X \to Y$ is stable then $tr(F) \in Y^X$.

3. Let $f,g : X \to Y$ be two stable functions.

Define $f \leq_B g$ (**Berry's order**) iff $\forall x,y \in X$ $x \subseteq y \Rightarrow f(x) = f(y) \cap g(x)$

Prove that $f \leq_B g$ if and only if $Tr(f) \subseteq Tr(g)$. Let moreover $\leq_p$ be the pointwise order. Prove that:

i. $f \leq_B g \Rightarrow f \leq_p g$

ii. $f \uparrow g \Rightarrow (f \leq_B g \Leftrightarrow f \leq_p g)$

4. Let X,Y be coherent domains. A stable function $f: X \to Y$ is **linear** iff :

i. $a \cup b \in X \Rightarrow f(a \cup b) = f(a) \cup f(b)$

ii. $f(\emptyset) = \emptyset$

Prove that $f: X \to Y$ is linear iff its trace is formed of pairs (a,z), where the component a is a singleton. Observe that the maps f^+ and f^- in 2.4.2.4 are actually linear. Call **Lin** the category of coherent domains and linear maps.

5. Let $f: X \to Y$, $g: X \to Y$ be two linear functions. Prove that $Tr(f) \subseteq Tr(g)$ if and only if for all x in X $f(x) \leq g(x)$. Deduce as a corollary that on linear functions between coherent domains the order of Berry coincides with the pointwise order.

2.4.2.7 Definition *The function* $\boldsymbol{eval_{X,Y}}$*:* $Y^X \times X \to Y$ *is defined by the following equation:* $\forall A \in Y^X, \forall B \in X \ eval_{X,Y}(A,B) = \{y \ / \ \exists (b,y) \in A, \ b \subseteq B\}$.

We prove that $eval_{X,Y}$ is stable. Continuity is trivial. We must only check that if $(A,B) \uparrow (A',B')$ [mod Y^X] then $eval_{X,Y}((A,B) \cap (A',B')) = eval_{X,Y}((A,B)) \cap eval_{X,Y}((A',B'))$. The inclusion $\subseteq$ is immediate by continuity.

Take then z in $eval_{X,Y}((A,B)) \cap eval_{X,Y}((A',B'))$. This implies $\exists (b,z) \in A$, $b \subseteq B$ and $\exists (b',z) \in A'$, $b' \subseteq B'$. Note that $B \uparrow B'$ by hypothesis and, thus, $b \uparrow b'$. Moreover, also by hypothesis, $A \uparrow A'$ and then, by definition of consistency mod Y^X, one has $b = b'$. This implies $(b,z) \in A \cap A', b \subseteq B \cap B' \Leftrightarrow z \in eval_{X,Y}((A \cap A'),(B \cap B')) \Leftrightarrow z \in eval_{X,Y}((A,B) \cap (A',B'))$.

In conclusion, the category **Stab** of coherent domains and stable functions is a CCC.

Exercise Let f^+ and f^- be defined as in 2.4.2.4, over coherent domains X and Y. Prove that $f^- \circ f^+ = id_X$, i.e., that $X < Y$ via (f^+, f^-) in **Stab**.

By this technique and the following construction, one can easily construct, in each cardinal, a coherent domain of which all other coherent domains of the same cardinality are retracts. In particular, it will be so also its own function space.

2.4.2.8 Definition *If X is a coherent domain, then !X (read* ***of course*** *X) is the coherent domain defined by*

i. $|!X| = \{a \ / \ a \in X, \ a \text{ finite}\}$

ii. $a \uparrow b \ [mod \ !X]$ *iff* $a \cup b \in X$.

Let T be the three-element truth value poset {⊥,true, false}, i.e., the "lifting" of {true, false}. For simplicity, we look at the cardinal ω. Consider then the ω-power T^ω of T. That is, take all the functions from ω to T. When T is partially ordered in the usual way, with ⊥ least and true and false incomparable, then T^ω is clearly a coherent domain. One may also understand T^ω as the set of disjoint subsets of ω. Of course, $!T^\omega$ is aslo in **Stab** by the definition just given.

2.4.2.9 Theorem *Let D be a coherent domain with a countable* $|D|$*, and let* $e: \omega \to |D|$ *be a bijective map. Then there exist an injective function f from* $|X|$ *to* $|!T^\omega|$ *such that, for all* $x,x' \in |X|$*, one has* $\{x, x'\} \in X \Leftrightarrow \{f(x), f(x')\} \in !T^\omega$.

Proof Let $f: |D| \to |!T^\omega|$ be defined in the following way:

$$f(e(i)) = < \{i\}, \{j \,/\, j \leq i \text{ and } \textbf{not } e(i) \uparrow e(j)\} >$$

Obviously f is injective. It is also trivial that $\{e(i),e(j)\} \in D$ implies $\{f(e(i)),f(e(j))\} \in !T^\omega$. Conversely suppose **not** $e(i) \uparrow e(j)$ [mod D] , and let $i \leq j$ (the other case is analogous). This implies that $i \in f(e(j))_1$ and then **not** $f(e(i)) \uparrow f(e(j))$ [mod $!T^\omega$].

2.4.2.10 Corollary *Let* f^+ *and* f^- *be as in 2.4.2.4, for* $f: |D| \to |!T^\omega|$*. Then, for any coherent domain D with a countable* $|D|$*, one has* $D < !T^\omega$ *via* (f^+,f^-).

The proof easily follows from the exercise above. Note now that, if D is countably based, so is D^D, in **Stab** (check this for exercise). Since, in particular, $|!T^\omega|$ is countable, then $!T^\omega$ turns ou to be a reflexive object, as $!T^\omega !T^\omega <_ !T^\omega$.

2.5 Equalizers and Pullbacks

Let $f,g: A \to B$ a pair of "parallel" functions in **Set**, i.e. f, g have the same source and target. The subset E of A on which f and g agree, i.e., $E = \{x \,/\, x \in A, \text{ and } f(x) = g(x)\}$ is called an **equalizer** of f and g. We try now to give a categorical characterization of previous set-theoretic notion. The starting point is that E being a subset of A it must be represented as a subobject, that is, as a mono $i: E \to A$; moreover, i must enjoy the property $f \circ i = g \circ i$. But E is the maximal subset of A on which f and g agree, and in order to guarantee this condition we require that if $h: C \to A$ is any other function such that $f \circ h = g \circ h$, then h "factors" uniquely through i , that is, there exist a unique $k: C \to E$ such that $h = i \circ k$. We now prove that the previous condition is enough to ensure, in **Set**, that i(E) contains *all* the $x \in A$, such that $f(x) = g(x)$.

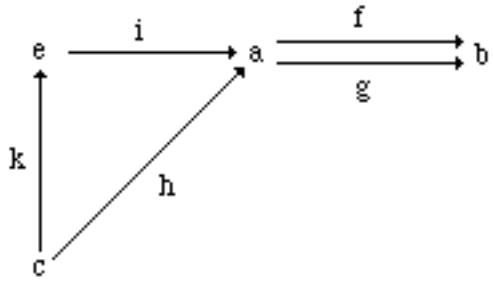
Suppose not, then there exist $a \in A$, $a \notin i(E)$ such that $f(a) = g(a)$. Consider the function $l: E \cup \{a\} \to A$ defined by $l(e) = i(e)$ if $e \in E$, $l(a) = a$. Of course $f \circ l = g \circ l$, and therefore a morphism $k: E \cup \{a\} \to E$ must exists such that $l = i \circ k$. But then $a = l(a) = i(k(a)) \in i(E)$; this is a contradiction.

Since this condition of unique factorization also implies that i is mono (see proposition 2.5.2 below) we are led to the following definition:

2.5.1 Definition *Given a pair of morphisms* $f,g \in \mathbf{C}[a,b]$, *an* ***equalizer*** *of* f *and* g *is a pair* $(e, i \in \mathbf{C}[e,a])$ *such that :*

i. $f \circ i = g \circ i$

ii. for all $h \in \mathbf{C}[c,a]$, $f \circ h' = g \circ h'$ *implies* $\exists!\, k \in \mathbf{C}[c,e]$ $i \circ k = h$.



Coequalizers *are defined dually, that is a coequalizer of* f,g *is a pair* $(e, i \in \mathbf{C}[b,e])$ *such that:*

i. $i \circ f = i \circ g$

ii. for all $h \in \mathbf{C}[b,c]$, $h \circ f = h \circ g$ *implies* $\exists!\, k \in \mathbf{C}[e,c]$ $k \circ i = h$.

2.5.2 Proposition *Every equalizer is monic.*

Proof Let i: e→a be the equalizer of f, g: a→b. Let j, l : c→e, such that $i \circ j = i \circ l$.
Since $f \circ (i \circ j) = (f \circ i) \circ j = (g \circ i) \circ j = g \circ (i \circ j)$ there exists a unique h: c→e such that $(i \circ j) = i \circ h$, hence $j = h = l$. ♦

2.5.3 Proposition *Every epic equalizer is iso.*

Proof Let i: e→a be the equalizer of f, g: a→b. Since i is epic, and $f \circ i = g \circ i$, it follows that f = g. The identity id_a equalizes f and g, and there is a unique morphism h: a→e such that $id_a = i \circ h$. Moreover $i \circ h \circ i = id_a \circ i = i \circ id_a$, and since i is monic (by proposition 2.5.2), then $h \circ i = id_a$. ♦

We now introduce one of the most powerful notions of Category Theory: the pullback. In a sense, pullbacks generalize equalizers to pairs of morphisms with different sources.

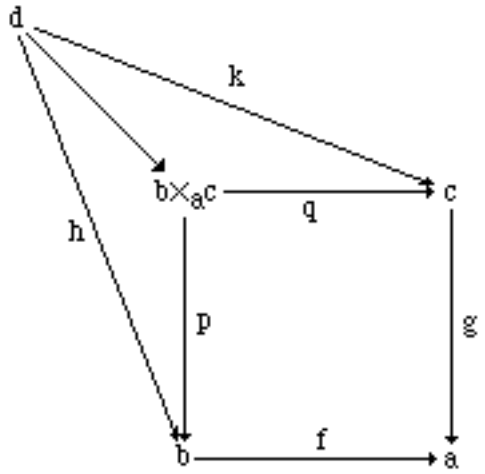
2.5.4 Definition *Given two arrows* $f: b \to a$ *and* $g: c \to a$ *with common target* a, *the* ***pullback*** *of* (f,g) *is an object* $b \times_a c$ *and two arrows* $p: b \times_a c \to b$, $q: b \times_a c \to c$, *such that*

1. $f \circ p = g \circ q: b \times_a c \to a$

2. for every other triple $(d, h: d \to b, k: d \to c)$ *such that* $g \circ k = f \circ h$, *there exists a unique arrow* $\langle h,k \rangle_a: d \to b \times_a c$ *such that* $p \circ \langle h,k \rangle_a = h$, *and* $q \circ \langle h,k \rangle_a = k$.

The dual notion is called ***pushout.***

A typical pullback diagram is as follows:



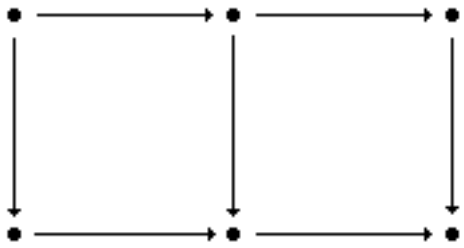
The lower "square" is also called "pullback square." Note that the notation used for pullbacks is quite similar to the one used for products; indeed, they behave similarly (products are just a particular case of pullbacks, see proposition. 2.5.5 below). Note also that the subscript a is meant to express the dependency of $b\times_a c$ and $\langle h,k\rangle_a$ on f and g, but $b\times_{f,g} c$ and $\langle h,k\rangle_{f,g}$, is too heavy a notation: the subscript must be considered essentially as a warning that we are dealing with a pullback, not just a product. Usually this omission of information is harmless, because the particular pullback we are considering is clear from the context.

Example In **Set** the pullback of (f: B→A, g: C→A) is as follows:
$(\{\langle x,y\rangle / x\in B, y\in C, f(x) = g(y)\}, p_1, p_2)$ where $p_1(\langle x,y\rangle) = x$ and $p_2(\langle x,y\rangle) = y$.

2.6.5 Proposition *Let **C** be a category with a terminal object t. For any object a of **C**, let !a be the unique morphism in **C**[a,t]. If **C** has pullbacks for every pair of arrows, then it also has products for every pair of objects.*
Proof *Hint*. Given a,b in C, let (a×b, p_1: a×b→a, p_2: a×b→b) be the pullback of (!a: a→t, !b: b→t). It is easy to verify that this is a product. ♦

2.6.6 Proposition *If a category **C** has pullbacks for every pair of arrows and it has terminal object, then it has an equalizer for every pair of arrows.*
Proof Let f,g: a→b. Let (c, fst:c→a, snd:c→a) be the pullback of ($\langle f,id_a\rangle$: a→b×a, $\langle g,id_a\rangle$: a→b×a). Then the equalizer of f,g is (c, fst = snd). Indeed, $f \circ fst = p_1 \circ \langle f\circ fst, fst\rangle = p_1 \circ \langle f, id_a\rangle \circ fst = p_2 \circ \langle g,id_a\rangle \circ snd = p_2 \circ \langle g\circ snd, snd\rangle = g \circ snd$. Moreover, for any (c', h:c'→a) such that $f \circ h = g \circ h$, also $\langle f,id_a\rangle \circ h = \langle g,id_a\rangle \circ h$; by definition of pullback, there exists a unque k: c'→c such that $fst \circ k = h$. ♦

2.6.7 Pullback Lemma (PBL) *If a diagram of the form*



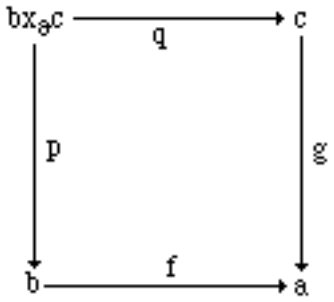
commutes, then

i. if the two small squares are pullbacks, then the outer rectangle is a pullback;

ii. if the outer rectangle and the right-hand square are pullbacks, then the left-hand square is a pullback.

Proof Exercise. ♦

2.6.8 Proposition *If the square*

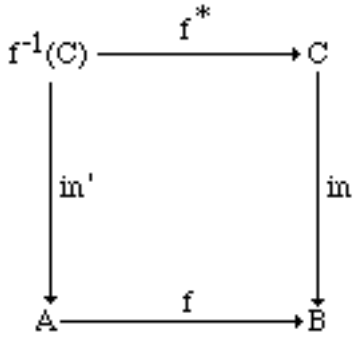


is a pullback and g is monic, then p is monic as well.

Proof: Exercise. ♦

The previous property suggests an interesting generalization of a common set-theoretic construction. If f is a function from a set A to a set B, and C is a subset of B, then the inverse image of C under f, denoted $f^{-1}(C)$ is that subset of A defined by $f^{-1}(C) = \{x/\ x \in A, f(x) \in C\}$.

It is easy to show that the diagram



is a pullback square in **Set**.

In general, given a monic g: c↣b and a morphism f: a→b, the **inverse image** of g under f is the subobject of a (if it exists) obtained by pulling back g along f.

2.6 Partial Morphisms and Complete Objects

As mentioned in the introduction, the common perspective of Programming Language Theory and Category Theory is due to the priority given to "functions" with respect to "sets." The latter notion, when required, is a derived notion of the former, in a sense.

There is an other aspect, though, that should be considered. In Computation Theory, as well as in actual programming, diverging computations cannot be avoided unless a restriction is made to a subclass of the computable functions.

The notion of partiality has a natural interpretation over sets. Let f: A→B be a partial function; the domain of convergence of f, call it $f{\downarrow}$, is just a subset of its "domain" A in the broader sense; moreover the restriction of f to $f{\downarrow}$ is a total map $f_{|(f{\downarrow})}: f{\downarrow} \to B$. Thus a partial map f may be represented by a pair of total functions (i: D→A, h: D→B), where D⊆A is the domain of convergence of f, h is the restriction of f to D, and i: D→A is the canonical injection .

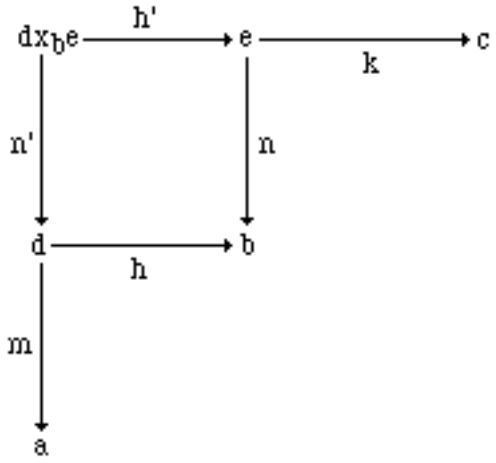
If we try to simulate the previous definition in categorical language, it is natural to define a partial map f between two objects a and b in a category **C** as a pair (m: d→a , h: d→b), where m is monic. However, as in the case of subobjects, we have no way to choose d in a canonical way, so we are forced to identify partial morphisms up to isomorphic variations of d (see section 1.5.)

2.6.1 Definition *Given a category **C** and two objects a and b, a partial map [m, h]: a→b is an equivalence class of pairs (m: d→a, h: d→b), where m is monic, with respect to the following relation R: (m: d→a, h: d→b) R (m': d'→a, h': d'→b) iff* $\exists k: d \to d'$*, k iso,* $m' = m \circ k$ *,* $h' = h \circ k$*.*

With a little abuse of language, we will often speak about a particular pair (m: d→a, h: d→b) when we actually mean $[(m: d \to a, h: d \to b)]_R$.

Our next aim is to define a category **pC** of partial maps on **C**. The main problem is in defining composition. If **C** has pullbacks for every pair of arrows, then the problem is resolved in the following way.

Given (n: e→b, k: e→c) and (m: d→a, h: d→b), define [n, k] ∘ [m, h]: a→b as the equivalence class determined by the outermost sides in the following diagram, i.e., ($m \circ n'$: $d \times_b e \to a$, $k \circ h'$: $d \times_b e \to c$) :



where the square is a pullback. Note that by proposition 2.5.8, since n is monic, n' is monic, too.

Let us see how the previous definition works in **Set**. For the equivalence relation defined on partial morphisms, we suppose $d \subseteq a$, $e \subseteq b$ and take $m: d \to a$, $n: e \to b$ as canonical injections. Then $d \times_b e = \{(x,y) / x \in d, y \in e, h(x) = n(y)\} = \{(x,y) / x \in d, y \in e, h(x) = y\} \cong \{x / x \in d, h(x) \in e\}$, that is, the expected domain of convergence of the composed function. The projections h' and n' associated with the pullback $\{x / x \in d, h(x) \in e\}$ are respectively $h_{|\{x / x \in d, h(x) \in e\}}$ and the canonical injection $i: \{x / x \in d, h(x) \in e\} \to d$. In conclusion, for all $x \in \{x / x \in d, h(x) \in e\}$, one has $k(h'(x)) = k(h(x))$, as we wanted.

Every arrow $f \in \mathbf{C}[a,b]$ has a natural associated arrow in **pC**, that is $(id: a \to a, f: a \to b)$. We say that a map in **pC** is **total** iff it has the above form (up to equivalences). $(\mathbf{pC})_\mathbf{t}$ is the subcategory of **pC** of total maps. As will become clearer in the next chapter, **C** is "isomorphic" to $(\mathbf{pC})_\mathbf{t}$.

Exercise Prove the following assertions:

1. id is total;
2. f,g total $\Rightarrow$ f∘g total;
3. f∘g total $\Rightarrow$ g total;

Example **pSet** is the category of sets with partial maps as morphisms. **pR** = **PR** is the monoid of the partial recursive functions. **pEN** is defined by using partial recursive functions in the diagram that defines the morphisms, instead of the total ones (see section 2.2). Observe that the diagrams commutes iff $e_b \circ f' = f \circ e_a$ for $f' \in \mathbf{PR}$ and, hence, $e_a(n) = e_a(m)$ and $f'(n)\downarrow \Rightarrow f'(m)\downarrow$. Clearly, $(\mathbf{pSet})_\mathbf{t} = \mathbf{Set}$, $\mathbf{PR}_\mathbf{t} = \mathbf{R}$ and $(\mathbf{pEN})_\mathbf{t} = \mathbf{EN}$.

Remark Sometimes, in the construction of the category **pC** from **C**, it may be intersting to restrict our attention to only a subset of the class of monics of **C**. Consider, for example, the category **PO** of p.o.sets: an interesting definition of a partial map f between two p.o.sets would also require f to be defined in an upward closed subset. Let **C** be a category with pullbacks for every pair of arrows. An

admissible family M of monics of **C**, is a family of monics closed under identities, compositions, and pullbacks. Every admissible M on a category **C** gives rise to a different category (M)(**pC**) of partial maps.

As already recalled, diverging computations play an essential role in the theory of computation. They also have an obvious interpretation in Set Theory. However, the set-theoretic understanding of computations may not suffice. Type-free languages and many typed ones, as we shall see later, escape a "naive" interpretation as sets and functions.

As a matter of fact, since the early days of denotational semantics of programming languages, the need to give meaning to (possibly) diverging computations over nontrivial mathematical structures suggested the introduction of various categories of p.o.sets with least elements. The naive understanding of this is immediate: add to all required data types, as sets, an extra element and give them a p.o.set structure as flat p.o.sets (i.e., $\perp$ is the least element and all others are incomparable). We already dealt with this in practice, when presenting various categories of p.o.sets as examples of CCC's. In a sense, the bottom element $\perp$ provides the first hint of the introduction of approximation and continuity notions to the mathematical semantics of programs.

As we have demonstrated, this concept is very clear matematically, at least in several specific categories, such as continuous or algebraic lattices, c.p.o's, and Scott domains. It is not so simple in interesting categories for computations such as **EN**, though. In order to understand it in a general setting and avoid any abuse of this simple concept, a category-theoretic perspective may provide a sound mathematical frame.

Notation In the rest of this section, **C** is a category of partial maps, i.e., **C = pD** for some category **D** with pullbacks for every pair of arrows. $\mathbf{C_t}$ is the associated category of total maps. Since $\mathbf{C_t}$ is a subcategory of **C**, we compose total and partial morphisms by using the same operation of composition. For typographical reasons, we write $a°$ instead of $a_\perp$.

The set-theoretic idea we try to formalize categorically is that, when an object a is "lifted" to $a°$ by adding an extra least element, then any hom-set of total maps with target $a°$ is isomorphic to the corresponding hom-set with target a.

2.6.2 Definition. *The **lifting** of $a \in Ob_{\mathbf{C}}$ is an object $a° \in Ob_{\mathbf{C_t}}$ together with a morphism $ex_a \in \mathbf{C}[a°,a]$ and for every $c \in Ob_{\mathbf{C}}$ an operation $\tau_c : \mathbf{C}[c,a] \to \mathbf{C_t}[c,a°]$ such that, for every $f \in \mathbf{C}[c,a]$ and for every $g \in \mathbf{C_t}[c,a°]$ one has*

1. $ex_a \circ \tau_c(f) = f$
2. $\tau_c(ex_a \circ g) = g$.

The operation τ_c: $\mathbf{C}[c,a] \to \mathbf{C_t}[c,a^\circ]$ is bijective, as τ_c^{-1} is defined by $\tau_c^{-1}(g) = ex_a \circ (g)$. Conversely, if τ_c is bijective and (1) holds, then also (2) does. Indeed, let $g \in \mathbf{C_t}[c,a^\circ]$; since τ_c is surjective, then there is $f \in \mathbf{C}[c,a]$ such that $\tau_c(f) = g$. That is, $\tau_c(ex_a \circ g) = \tau_c(ex_a \circ \tau_c(f)) = \tau_c(f) = h$ Thus, the following is an equivalent definition of the lifting object:

2.6.3 Definition. *The **lifting** of $a \in Ob_{\mathbf{C}}$ is an object $a^\circ \in Ob_{\mathbf{C}}$ together with a morphism $ex_a \in \mathbf{C}[a^\circ,a]$ such that, for every $f \in \mathbf{C}[c,a]$, there exists one and only one $g \in \mathbf{C_t}[c,a^\circ]$ satisfying the equation: $ex_a \circ g = f$.*

Thus, as we wanted, for all b, $\mathbf{C}[b,a]$ and $\mathbf{C_t}[b,a^\circ]$ are isomorphic, since any partial morphism $f \in \mathbf{C}[b,a]$ may be uniquely extended to a total one when the target object is lifted (and the lifting exists).

Exercise Prove that the lifting a° of an object a is unique, if it exists.

2.6.4 Proposition. *Let a° be the lifting of a and set $in_a = \tau_a(id_a)$. Then a is a retract of a° in $\mathbf{C}$ (notation: $a <_p a^\circ$) via (in_a, ex_a).*

Proof.
$$\begin{aligned} ex_a \circ in_a &= (\tau_{a^\circ})^{-1}(id) \circ in_a && \text{by definition of } (\tau_{a^\circ})^{-1} \\ &= (\tau_a)^{-1}(id \circ in_a) \\ &= (\tau_a)^{-1}(\tau_a(id)) \\ &= id. \ \blacklozenge \end{aligned}$$

2.6.5 Definition. *An object $a \in Ob_{\mathbf{C}}$ is a **complete** object iff $a < a^\circ$ in $\mathbf{C_t}$.*

The intuition should be clear. An object is complete when it "already contains," in a sense, the extra $\perp$. Think of an object d of **pCPO** and take its lifting d°, i.e., add a least element $\perp$ to d. Then d is complete, that is, $d < d^\circ$ via (i,j), iff d already contained a least element, $j(\perp)$ to be precise. It is actually easy to show that the complete objects in **pPO**, and likewise in **pCPO**, are exactly the partial ordered sets with a least element.

Note that in contrast to the partial retraction $a <_p a^\circ$ via (in,ex) in definition 2.6.4, which may be always given, only complete objects yield total retractions $a < a^\circ$.

2.6.6 Lemma. *If $a < a^\circ$ via (i,j) (in $\mathbf{C_t}$), then $\exists out \in \mathbf{C_t}[a^\circ,a]$ $a < a^\circ$ via $(\tau_a(id_a), out)$.*

Proof Set $out = j \circ \tau(ex_a \circ i \circ ex_a)$. Then

$$\begin{aligned} out \circ \tau_a(id_a) &= j \circ \tau(ex_a \circ i \circ ex_a) \circ \tau_a(id_a) \\ &= j \circ \tau(ex_a \circ \tau(ex_a \circ i \circ ex_a) \circ \tau_a(id_a)) && \text{by (2) in def. 2.6.2} \\ &= j \circ \tau(ex_a \circ i \circ ex_a \circ \tau_a(id_a)) && \text{by (1)} \end{aligned}$$

$= j \circ \tau(ex_a \circ i)$ by (1)
$= j \circ i$ by (2)
$= id_a$. ♦

Thus we may then assume that, if a is complete, $a < a^\circ$ via $(in=\tau_a(id_a),out)$.

The following fact gives the main motivation for the invention of complete objects: exactly on complete objects as targets all partial morphisms may be extended to total ones, *with the same target.*

2.6.7 Definition *$\underline{f} \in \mathbf{C}_t[b,a]$ **extends** $f \in \mathbf{C}[b,a]$ iff $\forall c \in Ob_{\mathbf{C}}$, $\forall h \in \mathbf{C}_t[c,b]$, $(f \circ h) \in \mathbf{C}_t[c,a] \Rightarrow \underline{f} \circ h = f \circ h$.*

2.6.8 Theorem. *Let a° be the lifting of $a \in Ob_{\mathbf{C}}$. Then $a < a^\circ \Leftrightarrow \forall b, \forall f \in \mathbf{C}[b,a], \exists \underline{f} \in \mathbf{C}_t[b,a]$ such that $\underline{f}$ extends f.*

Proof.($\Rightarrow$) Set $\underline{f} = out \circ \tau_b(f)$. Then, for every $h \in \mathbf{C}_t[c,b]$, such that $(f \circ h) \in \mathbf{C}_t[c,a]$,

$\underline{f} \circ h = out \circ \tau_b(f) \circ h$
$= out \circ \tau_c(ex_a \circ \tau_b(f) \circ h)$ by (2) in def. 2.6.2
$= out \circ \tau_c(f \circ h)$ by (1)
$= out \circ \tau_c(ex_a \circ \tau_a(id_a) \circ f \circ h)$ by (1)
$= out \circ \tau_a(id_a) \circ f \circ h$ by (2)
$= f \circ h$

($\Leftarrow$) just take $\underline{ex} \in \mathbf{C}_t[a^\circ,a]$. ♦

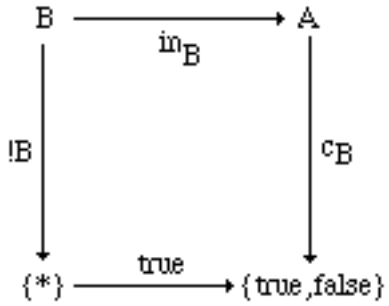
Exercises Prove the following facts:

1. $a^\circ < a^{\circ\circ}$.
2. $b < a < a^\circ \Rightarrow b < b^\circ$. (*Hint:* let $b < a$ via (i,j). For any $f \in \mathbf{C}[c,b]$ consider the extension $\underline{i \circ f} \in \mathbf{C}_t[c,a]$ of $i \circ f \in \mathbf{C}[c,a]$. Then $j \circ \underline{i \circ f} \in \mathbf{C}_t[c,b]$ and $\forall h$ into dom(f) $j \circ \underline{i \circ f} \circ h = j \circ (i \circ f \circ h) = f \circ h$.)

2.7 Subobject Classifiers and Topoi

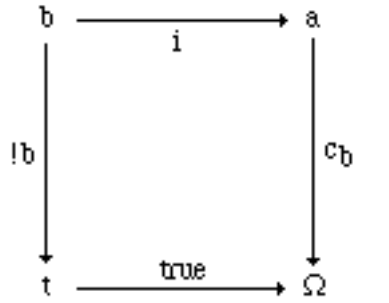
Subobjects and partial morphisms already forced some typical set-theoretic notions into the realm of categories. Let us take now a further step and categorically reconstruct power-sets and related constructions. By this, it will be possible to relativize these notions to structured sets. As for subobjects, the categorical version carries the structural information of the intended category.

In **Set** there is a one-to-one correspondence between subset of a set A and functions from A to 2={true, false}. The isomorphism takes every $B \subseteq A$ to its characteristic function $c_B: A \to 2$, defined by the following: $c_B(a)$ = true if $a \in B$, $c_B(a)$ = false if $a \notin B$. This isomorphism may be expressed in the categorical language by means of a pullback diagram



where in_B: B→A is the subset inclusion, and true: {*}→{true, false} takes * to "true."
The above diagram suggests the following definition in Category Theory:

2.7.1 Definition *Let* ***C*** *be a category with a terminal object t. A* ***subobject classifier*** *is an object* Ω *together with a morphism* ***true****: t→Ω such that for every monic i: b→a there is a unique arrow* c_b*: a→Ω such that the following diagram is a pullback diagram:*



Exercise A subobject classifier, when it exists in **C**, is unique up to isomorphisms.

The subobject classifier Ω plays a central role in the translation of Set Theory into Category Theory. It allows the simulation into the categorical language of concepts like intersection, union, and complement, by the definition of a Heyting algebra of truth-morphisms over Ω.

2.7.2 Definition *A* ***topos*** *is a category* ***C*** *with a terminal object, a subobject classifier, pullbacks for every pairs of arrows, and exponents for all pairs of objects.*

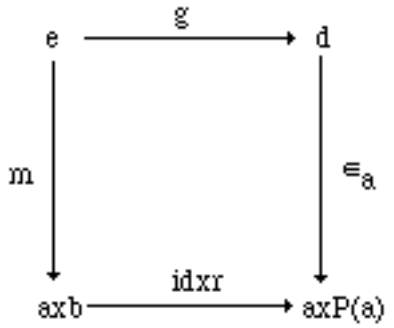
It will turn out that a topos is a "universe" where we can carry out constructions with almost the same confidence as we do in **Set**. Of course, **Set** itself is a topos.

Exercises

1. Prove that if **C** is a topos, then **C** is a CCC.
2. In a topos **C**, what is the arrow eval: $\Omega^a \times a \rightarrow \Omega$?

In the spirit of a categorical description of set-theoretic concepts, one may also talk in topoi of "relations", "powersets", and so forth. Quite generally, given objects a and b, a **relation** on a and b, is simply a monic r: d→a×b. In particular, then, one may consider power-objects P(a) as given by a relation $\in_a$: d→a×P(a) with the following universal property:

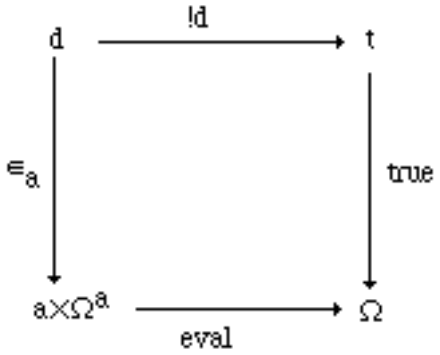
2.7.3 Definition *Let* ***C*** *be a cartesian category and* $a \in Ob_{\mathbf{C}}$. *The* ***power-object*** *P(a) is an object of* ***C*** *together with an object d and a monic* $\in_a$: *d→a×P(a) (a* ***membership-relation****) which is universal in the sense that, for any object b and monic m : e→a×b, there is a unique map r: b→P(a) and a (forcedly unique) map g: e→d to make the following square a pullback:*



As Ω is the truth value object, the set-theoretic intuition suggests that, in a topos, the object Ω^a should represent exactly the power-object of a.

2.7.4 Proposition *In a topos* ***C*** *every object a has a power-object* Ω^a.

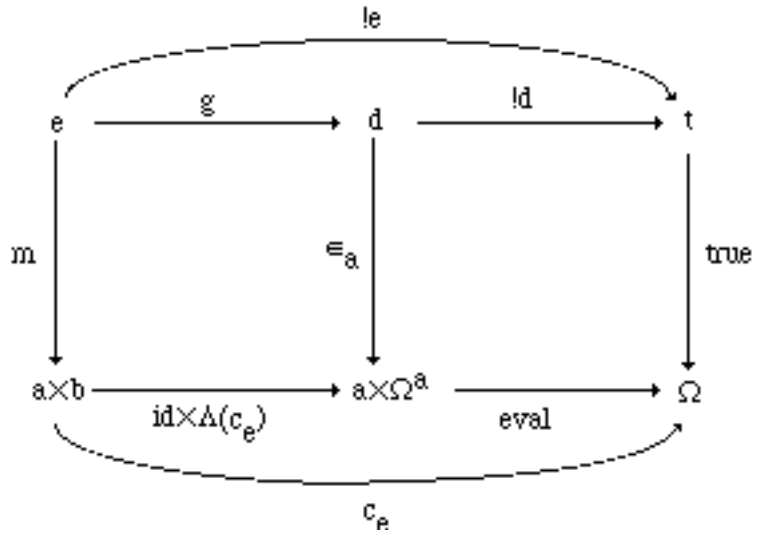
Proof. Let $\in_a$: d→a×P(a) be defined by the following pullback square:



Then the required universality of $\in_a$ is given by the properties

1. of the subobject classifier, i.e. the existence of the characteristic map c_e of m;
2. of the map eval , i.e. the existence and unicity of $\Lambda(c_e)$;
3. of the above pullback ;

an by an application of the pullback lemma 2.5.7. All this is described by the following commuting diagram, where the squares are pullbacks:



◆

The reader who has completed exercise (2) above may easily understand the intuitive meaning of the construction in 2.7.3 and compare it to his set-theoretic understanding. In particular, eval says whether "an element of a is in a given subset of a ."

Exercises

1. Prove that any topos has lifting.
2. Prove that a category **C** is a topos if and only if it has a terminal objects, and all pullbacks and powerobjects.

References The general notions can be found in the texts mentioned at the end of chapter 1, though their presentation and notation may be different. For example, in the case of CCC's, this is so also because these categories play a greater role in the categorical approach to Type Theory, or to denotational semantics, than in other applications of Category Theory. Notice that **Grp** and **Top** are not CCC's and consider that the origin of Category Theory is largerly indebted to algebraic geometry.

Applications of universal concepts from Category Theory to Programming Language Theory have been developed by several authors, in particular for program specification. For instance, the work by the ADJ group is explicitly based on universal conditions for specification (initial algebras of abstract data types; see Goguen et al. (1973/78)). An early insight in the connections between programs and categories may be found in Burstall and Thatcher (1974). All these aspects go beyond the limited aims of this book, which privileges "theories of functions" over "algebraic theories." Some more notions with an algebraic flavor may be found in chapter 4.

As for the examples we have provided, they originated entire areas of research. In particular, the category **EN** was first introduced by Malcev, as a general setting of Recursion Theory, and widely used in Ershov (1973/75). Scott domains and related categories are broadly treated in several places,

e.g. Scott (1981/82), Scott and Gunter (1989) or many books in denotational semantics. Scott (1976) presents Pω as a reflexive object and discusses categories of retractions. The (generalized) Myhill-Shepherdson theorem and related matters may be found in Scott (1976), Giannini and Longo (1984), Berger (1986), Rosolini (1986), and Longo (1988a), among others.

Coherent domains are given in Berry (1978) and used in Girard (1986), where linear maps are introduced (with some relevant consequences, see section 4.4).

There is a broad literature on categories with partial maps. We partly followed the approach in DiPaola and Heller (1984), and in Longo and Moggi (1984), where the notions of "complete object" and *partial* Cartesian Closed Category were introduced (and applied in Asperti and Longo (1987)). The properties of complete objects carry on also when using a more topos-theoretic perspective, as shown in Moggi (1988), which is devoted to a deeper insight into the concepts just sketched here. New results and surveys on categories with partial morphisms may be also found in Rosolini (1986), Moggi (1988a), Robinson and Rosolini (1988) and Curien and Obtulowicz (1988). Independently of the category-theoretic approach, Plotkin (1984) used the category **pCPO** for the purposes of denotational semantics.

Toposes originated by works of Lawvere and Tierney. A (difficult) introduction to Topos Theory may be found in Johnstone (1977). The reader may also consult Goldblatt (1979) or Barr and Wells (1985) on this subject.