Higher Order Containers

Thorsten Altenkirch[1], Paul Levy[2], and Sam Staton[3]

[1] University of Nottingham
[2] University of Birmingham
[3] University of Cambridge

Abstract. Containers are a semantic way to talk about strictly positive types. In previous work it was shown that containers are closed under various constructions including products, coproducts, initial algebras and terminal coalgebras. In the present paper we show that, surprisingly, the category of containers is cartesian closed, giving rise to a full cartesian closed subcategory of endofunctors. The result has interesting applications in generic programming and representation of higher order abstract syntax. We also show that while the category of containers has finite limits, it is not locally cartesian closed.

1 Introduction

Containers are a representation of datatypes, using a set of shapes S and a family of positions P indexed over shapes. The associated datatype is given by a choice of shape and an assignment of data to positions over that shape, clearly this is an endofunctor of **Set**. In previous work [AAG05,Abb03] it was shown that all strictly positive datatypes give rise to containers. To include nested inductive and coinductive definitions it was necessary to introduce n-ary containers, corresponding to n-ary functors.

Containers can be used to analyze generic constructions on datatypes without having to do induction over the syntax of datatypes. E.g. in [AAGM05] containers are used to study the notion of a derivative of a datatype.

Other applications of containers are related to container morphisms which are a concrete and complete representations of polymorphic functions, i.e. natural transformations, between datatypes. In [PGM08] this is exploited to derive theorems about polymorphic functions on lists.

The previous results can be stated in terms of properties of the category of containers: it is closed under products, coproducts and exponentiation with a set and the extension functor into sets is full and faithful. Recent work by the 3rd author [Sta08] on using higher order representations in generalized structured operational semantics raised the question wether the category of containers is cartesian closed. In the present paper we can answer this question positively.

As a simple example consider the functor $\Lambda \in \mathbf{Set} \to \mathbf{Set}$ which assigns to any set of variables the set of untyped lambda terms over this set. This functor can be specified as the inital solution to the following equation in the category of endofunctors

$$\Lambda \simeq \mathrm{Id} + \Lambda^2 + \mathrm{Id} \to \Lambda$$

Here Id is the identity functor, and $\to$ refers to the exponential of endofunctors (which may or may not exist). It turns out that this higher order equation is equivalent to

$$\Lambda \simeq \mathrm{Id} + \Lambda^2 + \Lambda \circ \mathrm{Maybe}$$

where $\mathrm{Maybe}\, X = 1 + X$. Indeed, this leads directly to the well-known representation of λ-terms as a nested datatype in Haskell

```
data Lam a = Var a | App (Lam a) (Lam a) | Lam (Maybe a)
```

which has been studied in [AR99,BP99].

The category of containers can be defined wrt. any locally cartesian closed category with coproducts. We are going to use the language of Type Theory which to develop our results, which is the internal language of locally cartesian closed categories. Hence the constructions presented here can be intuitively understood as taking place in a naive set theory.

A natural question is whether the category of containers itself is a model of Type Theory, i.e. locally cartesian closed. We are able to construct pullbacks if we assume that the ambient category has quotient types, corresponding to exact coequalizers. However, we can show that the right adjoint to pullback doesn't exist in general.

2 Preliminaries

We work in an extensional Type Theory [ML74] as the internal langauge of locally cartesian closed categories with disjoint coproducts.

Set We use **Set** to denote our universe of small types we identfy families with functions into **Set**.

$0, 1$ An empty type $0 \in \mathbf{Set}$ and a unit type $1 \in \mathbf{Set}$. Categorically, those correspond to initial and terminal objects. We write $() \in 1$ for the unique inhabitant of 1 and $!_A \in A \to 1$ with $!_A\, a = ()$ for the unique map into 1.

Σ- and Π-types Given $A \in \mathbf{Set}$ and $B \in \mathbf{Set}$ given that $x \in A$ then $\Sigma x \in A.B, \Pi x \in A, B \in \mathbf{Set}$. Elements of Σ-types are pairs, if $a \in A$ and $b \in B[x := a]$ then $(a, b) \in \Sigma x \in A.B$, while elements of Π-types are functions: given $b \in B$ assuming $x \in A$ then $\lambda x.b \in \Pi x \in A.B$.

Equality types Given $a, b \in A \in \mathbf{Set}$ we write $a = b \in \mathbf{Set}$ for the equality type. The constructor for equality is reflexivity $\mathrm{refl}\, a \in a = a$ if $a \in A$.

2 A type of Booleans $0, 1 \in 2 \in \mathbf{Set}$, which is disjoint, i.e. we have that $(0 = 1) \to 0$ is inhabited.

We omit a detailed treatment of eliminators and use functional programming notation as present in Agda and Epigram. All our definitions can be translated into using the standard eliminators at the price of readability. To avoid clutter we adopt the usual type-theoretic device of allowing hidden arguments, if they are inferable from the use. We indicate hidden arguments by subscripting the type, i.e. writing $\Pi_{x \in A} B$ and $\Sigma_{x \in A} B$ instead $\Pi x \in A.B$ and $\Sigma x \in A.B$.

While finite products arise as non-dependent Σ-types, finite coproducts can be represented as

$$A_0 + A_1 = \Sigma b \in 2.\text{if } b \text{ then } A_1 \text{ else } A_0$$

We use the injections $\text{in}_i \in A_i \to A_0 + A_1$ with $\text{in}_i\, a = (i, a)$ for $i \in 2$.

Σ-types can also be used to encode set-comprehension. If the family $B \in \mathbf{Set}$ (given $a \in A$) is propositional, i.e. there is at most one element in B for any $a \in A$, we write $\{a \in A \mid B\}$ for $\Sigma a \in A.B$.

We are going to use type-theoretic representations of categorical concepts. Given a bifunctor $F : \mathbf{Set}^{\text{op}} \to \mathbf{Set} \to \mathbf{Set}$ we define its end as a subset of the polymorphic functions:

$$\begin{aligned}
&\int_X F\, X\, X \in \mathbf{Type} \\
&\int_X F\, X\, X = \{f \in \Pi X \in \mathbf{Set}.F\, X\, X \\
&\qquad \mid \forall A, B \in \mathbf{Set}, g \in A \to B.F\, g\, B\, (f\, B) = F\, A\, g\, (f\, A)\}
\end{aligned}$$

This just internalizes the categorical definition of an end as a universal wedge. Dually, coends can be defined as quotients of large Σ-types (i.e. abstract datatypes), but we shall not need this here.

Using this notation, the type of natural transformations between two endofunctors $F, G \in \mathbf{Set} \to \mathbf{Set}$ arises as $\int_X F\, X \to G\, X$. The Yoneda lemma becomes:

$$F\, X \simeq \int_Y (X \to Y) \to F\, Y \qquad \text{Yoneda}$$

As it is well known the category of endofunctors has products which can be calculated pointwise:

$$(F \times G)\, X = F\, X \times G\, X$$

If we assume that the exponential of two endofunctors $F \to G$ exists then it must have a certain form reminiscent of the Kripke interpretation of implication:

$$\begin{aligned}
&(F \to G)\, X \\
&\simeq \int_Y (X \to Y) \to (F \to G)\, Y \qquad \text{Yoneda} \\
&\simeq \int_Y (X \to Y) \times F\, Y \to G\, Y \qquad \text{adjoint} \\
&\simeq \int_Y (X \to Y) \to F\, Y \to G\, Y \qquad \text{curry}
\end{aligned} \tag{1}$$

However, for size reasons $F \to G$ doesn't always exist. E.g. in the category of classical sets, which after all is a model of Type Theory, we have that the collection of $\int_X \mathcal{P}\, X \to \mathcal{P}\, (\mathcal{P} X)$, where $\mathcal{P}$ is the covariant powerset functor, is not a set. Indeed, there is a natural transformation $\alpha_\kappa \in \int_X \mathcal{P}\, X \to \mathcal{P}\, (\mathcal{P} X)$ for every cardinal κ, where $\alpha_\kappa\, X\, S = \{T \subseteq S \mid \text{card}\, T < \kappa\}$ for every set X and $S \subseteq X$.

3 Containers

A container is given by a set of shapes $S \in \mathbf{Set}$ and a family of positions indexed over shapes $P \in S \to \mathbf{Set}$ - we write $S \lhd P$. We shall also use $\lhd$ as a binder, writing $s \in S \lhd P$ for $S \lhd \lambda s.P$. A container represents an endofunctor

$$\begin{aligned}&[\![S \lhd P]\!] \in \mathbf{Set} \to \mathbf{Set}\\ &[\![S \lhd P]\!]\, X = \Sigma s \in S.P\, s \to X\end{aligned}$$

Given containers $S \lhd P$ and $T \lhd Q$ a morphism $f \lhd r$ is given by

$$\begin{aligned}&f \in S \to T\\ &r \in \Pi_{s \in S} Q\,(f\, s) \to P\, s\end{aligned}$$

This constitutes the category of containers **Cont** with the obvious definitions of identity and composition. $[\![-]\!]$ extends to a functor $[\![-]\!] \in \mathbf{Cont} \to (\mathbf{Set} \to \mathbf{Set})$ assigning natural transformations to container morphisms:

$$\begin{aligned}&[\![f \lhd r]\!] \in \int_X [\![S \lhd P]\!]\, X \to [\![T \lhd Q]\!]\, X\\ &[\![f \lhd r]\!]\, X\, (s, p) = (f\, s, p \circ r)\end{aligned}$$

Indeed **Cont** gives rise to a full subcategory of the category of endofunctors as shown in [AAG05]:

Proposition 1. $[\![-]\!] \in \mathbf{Cont} \to (\mathbf{Set} \to \mathbf{Set})$ *is full and faithful.*

Containers also give rise to two modalities which operate on families: given $B \in A \to \mathbf{Set}$ we have

$$\begin{aligned}&\Box_{S \lhd P}\, B, \Diamond_{S \lhd P}\, B \in [\![S \lhd P]\!]\, A \to \mathbf{Set}\\ &\Box_{S \lhd P}\, B\, (s, h) = \Pi p \in P\, s.B\, (h\, p)\\ &\Diamond_{S \lhd P}\, B\, (s, h) = \Sigma p \in P\, s.B\, (h\, p)\end{aligned}$$

$\Box$ can be defined for any functor because it corresponds to applying the functor to the representation of the family as an arrow.

The identity functor is given by $\mathrm{Id} = 1 \lhd 1$ and given $S \lhd P$ and $T \lhd Q$ we can construct their composite:

$$(S \lhd P) \circ (T \lhd Q) = [\![S \lhd P]\!]\, T \lhd \Diamond_{S \lhd P} Q$$

Composition is functorial but we shall not explore the 2-categorical structure of **Cont** any further.

In [AAG05] it is shown that containers are closed under finite products and coproducts. Indeed they are closed under arbitrary products and coproducts. Given a family of containers $F \in I \to \mathbf{Cont}$ this family can be isomorphically presented as

$$\begin{aligned}&\boldsymbol{S} \in I \to \mathbf{Set}\\ &\boldsymbol{P} \in \Pi_{i \in I} S\, i \to \mathbf{Set}\end{aligned}$$

with $F\,i = \boldsymbol{S}\,i \lhd \boldsymbol{P}_i$. We write $\boldsymbol{S} \lhd \boldsymbol{P}$ for this family. We now define the coproduct and the product of $\boldsymbol{S} \lhd \boldsymbol{P}$:

$$\Sigma(\boldsymbol{S} \lhd \boldsymbol{P}) = (i, s) \in \Sigma i \in I.\boldsymbol{S}\,i \lhd \boldsymbol{P}\,s$$
$$\Pi(\boldsymbol{S} \lhd \boldsymbol{P}) = f \in \Pi i \in I.\boldsymbol{S}\,i \lhd \Sigma i \in I.\boldsymbol{P}\,(f\,i)$$

We summarize the operations on containers:

Proposition 2. *Containers contain and are closed under:*

identity

$$[\![\mathrm{Id}]\!]\,A \simeq A$$

composition

$$[\![(S \lhd P) \circ (T \lhd Q)]\!] \simeq [\![S \lhd P]\!] \circ [\![T \lhd Q]\!]$$

coproducts

$$[\![\Sigma\,(\boldsymbol{S} \lhd \boldsymbol{P})]\!]\,A \simeq \Sigma i \in I.[\![\boldsymbol{S}\,i \lhd \boldsymbol{P}_i]\!]$$

products

$$[\![\Pi\,(\boldsymbol{S} \lhd \boldsymbol{P})]\!]\,A \simeq \Pi i \in I.[\![\boldsymbol{S}\,i \lhd \boldsymbol{P}_i]\!]$$

It is important to realize that the infinite coproducts and products are internal wrt. to the ambient category. The case of constant exponentiation in [AAG05] arises as a constant product.

4 Containers are cartesian closed

Our central observation is that exponentiation with a container which has only one shape $1 \lhd P$, i.e. a container representing an internal hom functor $[\![1 \lhd P]\!]\,X = P \to X$, is straightforward.

$$\begin{aligned}
&([\![1 \lhd P]\!] \to F)\,X \\
&\simeq \int_Y (X \to Y) \to [\![1 \lhd P]\!]\,Y \to F\,Y && \text{using (1)} \\
&= \int_Y (X \to Y) \to (P \to Y) \to F\,Y \\
&\simeq \int_Y (X \to Y) \times (P \to Y) \to F\,Y && \text{uncurry} \\
&\simeq \int_Y (X + P \to Y) \to F\,Y && \text{adjunction} \\
&\simeq F(X + P)
\end{aligned}$$

To summarize we have that

$$[\![1 \lhd P]\!] \to F \simeq F \circ (+P) \tag{2}$$

where $(+P)\,X = X + P$. Extensionally, every container is the a coproduct of hom containers:

$$\begin{aligned} &[\![S \lhd P]\!]\,X \\ &\simeq \Sigma s \in S.P\,s \to X \\ &\simeq \Sigma s \in S.[\![1 \lhd P\,s]\!]\,X \end{aligned} \tag{3}$$

Because of proposition 1 this is also true in the category of containers;

$$S \lhd P \simeq \Sigma s \in S.1 \lhd P\,s \tag{4}$$

Combining these observations we can see that exponentiation by a container is always possible and can be constructed using products and composition:

$$\begin{aligned} &[\![S \lhd P]\!] \to F \\ &\simeq [\![\Sigma s \in S.1 \lhd P\,s]\!] \to F && \text{using (4)} \\ &\simeq \Pi s \in S.[\![1 \lhd P\,s]\!] \to F && \text{adjunction} \\ &\simeq \Pi s \in S.F \circ (+P\,s) && \text{using (2)} \end{aligned}$$

Proposition 3. *Given a container $S \lhd P$ and a functor $F \in \mathbf{Set} \to \mathbf{Set}$ we have:*

$$[\![S \lhd P]\!] \to F \simeq \Pi s \in S.F \circ (+P\,s)$$

Using proposition 2 we know that if F is a container then $\Pi s \in S.F \circ (+P\,s)$ is a container. Since containers are a full subcategory of $\mathbf{Set} \to \mathbf{Set}$ (prop. 1) this implies our main result:

Corollary 1. *The category of containers is cartesian closed, and the embedding $[\![-]\!] \in \mathbf{Cont} \to (\mathbf{Set} \to \mathbf{Set})$ preserves the cartesian closed structure.*

We can spell out the construction of the exponential by expanding the definitions of the operations involved. Note that $+P\,s$ is given by $l : 1 + P\,s \lhd l = \mathrm{in}_0\,()$:

$$\begin{aligned} &S \lhd P \to T \lhd Q \\ &\simeq \Pi s \in S.(T \lhd Q) \circ (+P\,s) \\ &\simeq \Pi s \in S.(T \lhd Q) \circ (l : 1 + P\,s \lhd l = \mathrm{in}_0\,()) \\ &\simeq \Pi s \in S.[\![T \lhd Q]\!](1 + P\,s) \lhd \Diamond_{T \lhd Q}(\lambda l.l = \mathrm{in}_0\,()) \\ &\simeq f \in \Pi s \in S.[\![T \lhd Q]\!](1 + P\,s) \lhd \Sigma s \in S.\Diamond_{T \lhd Q}(\lambda l.l = \mathrm{in}_0\,())\,(f\,s) \\ &\simeq f \in \Pi s \in S.\Sigma t \in T.Q\,t \to 1 + P\,s \\ &\quad \lhd \Sigma s \in S.\Sigma q \in Q\,s.(f\,s).2\,q = \mathrm{in}_0\,() \end{aligned}$$

5 Failure of local cartesian closure

The previous section shows that we can interpret the simply typed λ-calculus within the category of containers. Can we go further, i.e. can we interpret dependent types in **Cont**?

Dependent types correspond to locally cartesian closed categories, LCCCs. A category is locally cartesian closed, if it has a terminal object and pullbacks (i.e. all finite limits) and a (fibred) right adjoint to the pullback functor. We will show that pullbacks can indeed be constructed, if we have quotient types (i.e. exact coequalizers) in the underlying Type Theory. However, we can also show that the right adjoints do not exist in general and hence while the category of containers has finite limits, it is not locally cartesian closed.

We know from the previous section that **Cont** has a terminal object $1 \lhd 0$ because this is just the nullary case of a finite product. Pullbacks correspond to products in the slice category, i.e. given

$$f_i \lhd a_i \in \mathbf{Cont}\,(S_i \lhd P_i)\,(T \lhd Q) \qquad\qquad i \in 2$$

we need to construct a new container $U \lhd R = (f_0 \lhd a_0) \times_{T \lhd Q} (f_1 \lhd a_1)$ together with projections:

$$g_i \lhd h_i \in \mathbf{Cont}\,(U \lhd R)\,(S_i \lhd P_i) \qquad\qquad i \in 2$$

We define

$$\begin{aligned}
&U \in \mathbf{Set}\\
&U = \{(s_0, s_1) \in S_0 \times S_1 \mid f_0\, s_0 = f_1\, s_1\}\\
&R \in U \to \mathbf{Set}\\
&R\,(s_0, s_1) = (P_0\, s_0 + P_1\, s_1)/\sim
\end{aligned}$$

where $\sim$ is an equivalence relation on $P_0\, s_0 + P_1\, s_1$ generated by

$$\mathrm{in}_0\,(a_0\, q) \sim \mathrm{in}_1\,(a_1\, q)$$

for $q \in Q\,(f_0\, s_0)$ which due to the assumption $f_0\, s_0 = f_1\, s_1$ is equivalent to $q \in Q\,(f_1\, s_1)$. The definition of $g_i \lhd h_i$ of the projections is straightforward : $g_i \lhd h_i = \pi_i \lhd \mathrm{in}_i$.

Proposition 4. *$U \lhd R, g_i \lhd h_i$ is a pullback of $f_i \lhd a_i$ in* **Cont**.

We omit the laborious verification of the equational conditions. As a consequence we have:

Corollary 2. Cont *has all finite limits.*

The restriction to finite limits isn't essential, it is not hard to generalize the construction to arbitrary limits, again in the appropriate internal sense.

Pullbacks are products in the slice category, i.e. for a given container A the slice category $\mathbf{Cont}/A$ has as objects morphisms with codomain A and as morphisms commuting triangles. Given arrows $\alpha, \beta \in \mathbf{Cont}/A$ their pullback $\alpha \times_A \beta$ is the product in $\mathbf{Cont}/A$. For the category to be locally cartesian closed we need a right adjoint to $\times_A$: assume $\alpha, \beta, \delta \in \mathbf{Cont}/A$, there is a $\beta \to_A \delta \in \mathbf{Cont}/A$ such that

$$\mathbf{Cont}/A\,(\alpha \times_A \beta)\,\delta \simeq \mathbf{Cont}/A\,\alpha\,(\beta \to_A \delta)$$

There is the additional condition that the local exponentials are fibred. However, this doesn't matter here because we are going to construct a counterexample, showing that already the isomorphism above doesn't exist in general. We set

$$\begin{aligned} &A = 1 \lhd 2 \\ &\alpha = \delta = id_A \in \mathbf{Cont}\, A\, A \\ &\beta = !_1 \lhd !_2 \in \mathbf{Cont}\, \mathrm{Id}\, A \qquad\qquad \text{there is only one} \end{aligned}$$

The pullback $\alpha \times_A \beta$ is again β. There is only one morphism in $\mathbf{Cont}/A\,\beta\,\delta$ since there is only one in $\mathbf{Cont}\,\mathrm{Id}\,A$. Now assume that $\beta \to_A \delta$ exists, i.e. let $\beta \to_A \delta = !_S \lhd f \in \mathbf{Cont}\,(S \lhd P)\,A$ with $f \in \Pi s \in S.2 \to P\,s$. Because of the isomorphism there can only be one morphism in $\mathbf{Cont}/A\,\alpha\,(\beta \to_A \delta)$. This allows us to draw some conclusions about the exponential $(!_S \lhd f)$:

1. There is a $t \in S$ and $g \in P\,t \to 2$ such that $g \circ (f\,t) = \mathrm{id}_2$.
2. For all $s \neq t$ $f\,s$ is not injective, i.e. $f\,t\,0 = f\,t\,1$.

Now consider $\gamma = ! \lhd \mathrm{in}_0 \in \mathbf{Cont}\,(2 \lhd 2 + 1)\,A$. There are two morphisms $\mathbf{Cont}/A\,(\gamma \times_A \beta)\,\delta$. However using the conditions about $\beta \to_A \delta$ above we can show that there is only one morphism in $\mathbf{Cont}/A\,\gamma\,(\beta \to_A \delta)$. Hence, **Cont** cannot be locally cartesian closed.

6 Conclusions and further work

The category of containers is a full subcategory of the category of endofunctors with a number of interesting closure properties. The initial motivation was to find a subcategory which is closed under taking initial algebras and terminal coalgebras and has the necessary infrastructure to define datatypes, i.e. products and coproducts. The fact that this category is also cartesian closed is an additional benefit and shows that we can interpret higher order constructions. Finite limits are also an interesting feature which may help in modelling dependent types directly, however, the failure of local cartesian closure might indicate that we should look for a different category. Quotient containers [AAGM04] might be an interesting alternative but an initial analysis indicates that they are not locally cartesian closed either. However, the failure of local cartesian closure may not be such an issue since dependent types can be modelled using indexed containers [AM09].

It is usual to work in an ambient category in which initial algebras of containers exist (W-types). However, the container Λ for λ-terms, in the introduction, is an initial algebra of a container, not in the category of sets, but in the category of containers. An investigation into the nature of W-types in the category of containers is left for future work.

References

[AAG05] M. Abbott, T. Altenkirch, and N. Ghani. Containers - constructing strictly positive types. *Theoretical Computer Science*, 342:3–27, September 2005. Applied Semantics: Selected Topics.

[AAGM04] M. Abbott, T. Altenkirch, N. Ghani, and C. McBride. Constructing polymorphic programs with quotient types. In *7th International Conference on Mathematics of Program Construction (MPC 2004)*, 2004.

[AAGM05] M. Abbott, T. Altenkirch, N. Ghani, and C. McBride. ∂ for data: derivatives of data structures. *Fundamenta Informaticae*, 65(1,2):1–128, March 2005.

[Abb03] M. Abbott. *Categories of Containers*. PhD thesis, University of Leicester, 2003.

[AM09] T. Altenkirch and P. Morris. Indexed containers. *submitted for publication*, 2009.

[AR99] T. Altenkirch and B. Reus. Monadic presentations of lambda terms using generalized inductive types. In *Computer Science Logic*, 1999.

[BP99] R. Bird and R. Paterson. Generalised folds for nested datatypes. *Formal Aspects of Computing*, 11(3), 1999.

[ML74] P. Martin-Löf. An intuitionistic theory of types: Predicative part. In H. E. Rose and J. C. Shepherdson, editors, *Proceedings of the Logic Colloquium*, pages 73–118. North-Holland, 1974.

[PGM08] R. Prince, N. Ghani, and C. McBride. Proving Properties of Lists using Containers. In *FLOPS*, 2008.

[Sta08] S. Staton. General structural operational semantics through categorical logic. *Symposium on Logic in Computer Science*, pages 166–177, 2008.