Special issue: Advanced Programming Techniques for Construction of Robust, General and Evolutionary Programs

Research Paper

Distributive laws of directed containers

Danel AHMAN[1] and Tarmo UUSTALU[2]
[1]*Laboratory for Foundations of Computer Science, University of Edinburgh*
[2]*Institute of Cybernetics, Tallinn University of Technology*

ABSTRACT

Containers are an elegant representation of a wide class of datatypes in terms of positions and shapes. We have recently introduced directed containers as a special case to account for the common situation where every position in a shape determines another shape, informally the subshape rooted by that position. While containers interpret into set functors via a fully faithful functor, directed containers denote comonads fully faithfully. In fact, directed containers correspond to exactly those containers that carry a comonad structure. Directed containers can also be seen as a generalization (a dependently typed version) of monoids.

While the category of containers (just as the category of set functors) carries a composition monoidal structure, directed containers (just as comonads) do not generally compose. In this paper, we develop a concept of a distributive law between two directed containers corresponding to that of a distributive law between two comonads and spell out the distributive-law based composition construction of directed containers. This turns out to generalize the Zappa-Szép product of two monoids.

KEYWORDS

directed containers, comonads, distributive laws, monoids, Zappa-Szép products, mathematical structures in functional programming, dependently typed programming

1 Introduction

The containers of Abbott, Altenkirch and Ghani [1] are an elegant representation of a wide class of datatypes in terms of shapes and positions in shapes. They make a kind of "syntax" for programming and reasoning about these datatypes (along with the related proposal, the polynomial functors of Gambino and Hyland [11]).

In previous work [3], we have introduced directed containers as a special case of containers to account for the common situation where every position in a shape determines another shape, informally the subshape rooted by that position. Some examples are the datatypes of non-empty lists and node-labelled trees together with the corresponding zipper datatypes. While containers interpret into set functors via a fully faithful monoidal functor, directed containers interpret fully-faithfully into comonads. Furthermore, directed containers correspond to exactly those containers whose interpretation carries a comonad structure. Directed containers with exactly one shape are the same as monoids. General directed containers are a curious generalization of monoids, a dependently typed version.

Directed containers do not admit all constructions that containers do. As with comonads, it is straightforward to form a coproduct of two arbitrary directed containers, but explicit constructions of the product are complicated and depend on assumptions. It is well-known that a sufficient condition for the composition of the underlying functors of two comonads to again exhibit a comonad structure is the presence of a distributive law between these comonads.

In this paper, we develop a representation of distribu-

Received June 10, 2012; Revised November 6, 2012; Accepted December 22, 2012.
[1)] *d.ahman@ed.ac.uk*, [2)] *tarmo@cs.ioc.ee*
DOI: 10.2201/NiiPi.2013.10.2

tive laws between comonads whose underlying functor is a container. We use it to define a direct composition construction of directed containers. We discover that this construction generalizes the Zappa-Szép product of two monoids (known also as the knit product, general product, bicrossed product, bilateral semidirect product and studied in group theory and semigroup theory, for an introduction, see [7]). We then provide a number of examples of this construction, concentrating on the datatypes of streams and non-empty lists. Among those are several that we could only find after we had acquainted ourselves with some facts about Zappa-Szép products of monoids.

Comonads can be used to model notions of context-dependence in computation, the central examples being dataflow computation, attribute evaluation and cellular automata [20]. In such applications, composition of comonads corresponds to combining different notions of context. In this paper, we do not attempt to elaborate these applications and focus only on basic theory. But we want to convey our excitement about ever newer mathematical structures that emerge in functional programming.

The paper is organized as follows. First, we briefly review containers and their interpretation as set functors, emphasizing the composition monoidal structure that is present on the category of containers. Then we proceed to directed containers and their interpretation as comonads. We explain, in particular, that directed containers characterize those containers whose interpretation carries a comonad structure. In the main sections of the paper, we discuss distributive laws between directed containers and composition of directed containers based on a distributive law, and present our examples.

The talk of T. Uustalu at the Shonan meeting on dependently typed programming was on the basic theory of directed containers. Following the meeting, we presented the material at FoSSaCS 2012 and published it in the proceedings [3]. This paper is based on additional material on distributive laws of directed containers, presented at CMCS 2012 as a short talk, but not published formally until now.

2 Containers

We begin by a very brief summary of the basics of containers [1] as a representation for a wide class of (parameterized) datatypes.

A *container* $S \lhd P$ is given by a set S (of shapes) and a shape-indexed family $P : S \to \mathsf{Set}$ (of positions).

It is useful to think of the shapes as "templates" for datastructures and the positions as "blanks" in those templates that can be filled with data. For example, we can represent the datatype of streams as the container $S \lhd P$ with a single shape $S = 1$ and natural number positions $P * = \mathsf{Nat}$. The intuition is that a stream is nothing but an infinite sequence. The datatype of lists is represented by the container $S \lhd P$ whose shapes $S = \mathsf{Nat}$ are the possible lengths of lists and positions $P\, s = [0, s)$ provide s blanks for lists of length s. For non-empty lists, one takes $S = \mathsf{Nat}$ and $P\, s = [0, s]$.

A *morphism* between containers $S \lhd P$ and $S' \lhd P'$ is a pair $t \lhd q$ of maps $t : S \to S'$ (the shape map) and $q : \Pi\{s : S\}.\, P'\,(t\, s) \to P\, s$ (the position map). (Here and in the following, we use Agda's [16] syntax of braces for implicit arguments, i.e., for those arguments we may want to skip when they are inferrable from other arguments.) Note that positions are mapped "backwards": every position in the shape $t\, s$ returned by the shape map t is associated to some position in the given shape s. The *identity morphism* on any container $S \lhd P$ is given by $\mathsf{id}^{\mathrm{c}}\, \{S \lhd P\} = \mathsf{id}\, \{S\} \lhd \lambda\{s\}.\, \mathsf{id}\, \{P\, s\}$ and the *composition* of two container morphisms $t \lhd q$ and $t' \lhd q'$ by $(t \lhd q) \circ^{\mathrm{c}} (t' \lhd q') = t \circ t' \lhd \lambda\{s\}.\, q'\, \{s\} \circ q\, \{t'\, s\}$.

Containers form a category **Cont**.

The *interpretation* of a container $S \lhd P$ is the set functor $[\![S \lhd P]\!]^{\mathrm{c}}$ given by

- $[\![S \lhd P]\!]^{\mathrm{c}}\, X = \Sigma s : S.\, P\, s \to X$,
- $[\![S \lhd P]\!]^{\mathrm{c}}\, f\, (s, v) = (s, f \circ v)$.

Intuitively, elements of $[\![S \lhd P]\!]^{\mathrm{c}}\, X$ consist of a shape and an assignment of data from X to all positions in the shape. The stream and list containers denote the expected datatypes. For example, for the stream container, we have $[\![S \lhd P]\!]^{\mathrm{c}}\, X = \Sigma * : 1.\, \mathsf{Nat} \to X \cong \mathsf{Nat} \to X \cong \mathsf{Str}\, X$. Similarly, for the list container, one gets $[\![S \lhd P]\!]^{\mathrm{c}}\, X = \Sigma s : \mathsf{Nat}.\, [0, s) \to X \cong \mathsf{List}\, X$.

The interpretation of a container map $t \lhd q : S \lhd P \to S' \lhd P'$ is the natural transformation $[\![t \lhd q]\!]^{\mathrm{c}} : [\![S \lhd P]\!]^{\mathrm{c}} \to [\![S' \lhd P']\!]^{\mathrm{c}}$, given by

- $[\![t \lhd q]\!]^{\mathrm{c}}\, \{X\}\, (s, v) = (t\, s, v \circ q\, \{s\})$.

Interpretation makes a functor from **Cont** to [**Set**, **Set**]. This functor turns out to be fully faithful: for any two containers, interpretation of container morphisms between them is a bijection.

Very importantly for this paper, the category **Cont** carries a (composition) monoidal structure. The identity container is $\mathsf{Id}^{\mathrm{c}} = 1 \lhd \lambda *.\, 1$ (one shape and one position in it) whereas the composition of two containers is $(S_0 \lhd P_0) \cdot^{\mathrm{c}} (S_1 \lhd P_1) = \Sigma s : S_0.\, P_0\, s \to S_1 \lhd \lambda\, (s, v).\, \Sigma p_0 : P_0\, s.\, P_1\, (v\, p_0)$ (a shape is a pair of an outer shape and an assignment of an inner shape to every position in it; a position is a pair of an outer and an inner position). There are container isomorphisms

- $\rho : \forall\{C\}.\, C \cdot^{\mathrm{c}} \mathsf{Id}^{\mathrm{c}} \to C$,

- $\lambda : \forall\{C\}.\, \mathsf{Id}^{\mathrm{c}} \cdot^{\mathrm{c}} C \to C$ and
- $\alpha : \forall\{C\}\,\{C'\},\{C''\}.$
  $(C \cdot^{\mathrm{c}} C') \cdot^{\mathrm{c}} C'' \to C \cdot^{\mathrm{c}} (C' \cdot^{\mathrm{c}} C'')$

obeying Mac Lane's coherence conditions. The interpretation functor $[\![-]\!]^{\mathrm{c}} : \mathbf{Cont} \to [\mathbf{Set}, \mathbf{Set}]$ is monoidal: there are natural isomorphisms

- $\mathsf{e} : \mathsf{Id} \to [\![\mathsf{Id}^{\mathrm{c}}]\!]^{\mathrm{c}}$ and
- $\mathsf{m} : \forall\{C_0, C_1\}.\, [\![C_0]\!]^{\mathrm{c}} \cdot [\![C_1]\!]^{\mathrm{c}} \to [\![C_0 \cdot^{\mathrm{c}} C_1]\!]^{\mathrm{c}}$

satisfying the appropriate conditions.

3 Directed containers

Many datatypes exhibit natural additional structure. For example, each node in a stream or list defines a substream or sublist (suffix) rooted by that position; the same applies to node-labelled trees. When we represent such datatypes as containers, this additional structure manifests via subshapes determined by every position in a given shape. This additional structure is not considered in the theory of (general) containers. We will now summarize the central facts about directed containers [3], a specialization of containers concerned with exactly this additional structure.

A *directed container* is a container $S \lhd P$ together with three operations

- $\downarrow : \Pi s : S.\, P\, s \to S$ (the subshape given by a position),
- $\mathsf{o} : \Pi\{s : S\}.\, P\, s$ (the root position),
- $\oplus : \Pi\{s : S\}.\, \Pi p : P\, s.\, P\,(s \downarrow p) \to P\, s$ (translation of subshape positions into positions in the global shape),

satisfying the following two shape equations and three position equations:

1. $\forall\{s\}.\, s \downarrow \mathsf{o} = s$,
2. $\forall\{s, p, p'\}.\, s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'$,
3. $\forall\{s, p\}.\, p \oplus \{s\}\,\mathsf{o} = p$,
4. $\forall\{s, p\}.\, \mathsf{o}\,\{s\} \oplus p = p$,
5. $\forall\{s, p, p', p''\}.$
   $(p \oplus \{s\}\, p') \oplus p'' = p \oplus (p' \oplus p'')$.

Note that Equations 4–5 are only well-typed because of equations 1–2. For Equation 4 to type-check, the p in the l.h.s. must be an element of $P\,(s \downarrow \mathsf{o})$ whereas that on the right must belong to $P\, s$. For Equation 5 to type-check, the p'' on the left must belong to $P\,(s \downarrow (p \oplus p'))$ while that on the right must be of type $P\,((s \downarrow p) \downarrow p')$. Also note that we have adopted a custom infix notation for $\oplus$ by writing the implicit argument after the infix operation symbol.

Modulo the fact that the positions involved come from different sets, Equations 3–5 are the laws of a monoid. In the degenerate case $S = 1$, Equations 1–2 trivialize, we really have only one set of positions $P\,*$ and we get exactly a monoid. If S is general, but $s \downarrow p = s$ always, then each $P\, s$ is a monoid. (One might also notice that laws 1–2 bear similarity to the laws of a monoid action. If none of $P\, s$, $\mathsf{o}\,\{s\}$, $p \oplus \{s\}\, p'$ depends on s, then we have one single monoid and $\downarrow$ is then a right action of that monoid on S.)

Streams and non-empty list datatypes (with the substreams, sublists, i.e., suffixes, structure) are archetypal examples of directed containers. For example, the directed container structure on the container of streams is given by $* \downarrow p = *$, $\mathsf{o}\,\{*\} = 0$ and $p \oplus p' = p + p'$. For the non-empty list container, the directed container structure is given by $s \downarrow p = s - p$, $\mathsf{o}\,\{s\} = 0$ and $p \oplus p' = p + p'$.

But these are not the only directed container structures on these containers. E.g., in the case of streams, the set of positions in the single shape $*$ is Nat, which carries other monoid structures than the free monoid structure on one generator, 1.

Hence we also get a directed container when we define $\mathsf{o}\,\{*\} = 1$ and $p \oplus p' = p \times p'$. This corresponds to streams and sampling. Indeed: positions $0, 1, 2, \ldots$ of the "substream" determined by position p in a global stream translate to positions $0, p, 2 \times p, \ldots$ of the global stream. Or we could define $\mathsf{o}\,\{*\} = 0$ and $p \oplus p' = p$ max p'. We get a "padded" version of suffixes: positions $0, 1, 2, \ldots, p, p + 1, \ldots$ of the "substream" determined by position p in a global stream translate to its positions $p, p, p, \ldots, p, p + 1, \ldots$.

Similar structures are possible for non-empty lists where we get non-trivial definitions of $\downarrow$. For sampling, we must define $s \downarrow p = \mathsf{if}\ p = 0\ \mathsf{then}\ 0\ \mathsf{else}\ s \div p$, $\mathsf{o}\,\{s\} = \mathsf{if}\ s = 0\ \mathsf{then}\ 0\ \mathsf{else}\ 1$, $p \oplus p' = p \times p'$. And for padded suffixes, it is appropriate to define $s \downarrow p = s$, $\mathsf{o}\,\{s\} = 0$ and $p \oplus p' = p$ max p'.

A *morphism* between directed containers $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ and $(S' \lhd P', \downarrow', \mathsf{o}', \oplus')$ is a morphism $t \lhd q$ between the underlying containers $S \lhd P$ and $S' \lhd P'$ that satisfies the following equations:

1. $\forall\{s, p\}.\, t\,(s \downarrow q\, p) = t\, s \downarrow' p$,
2. $\forall\{s\}.\, \mathsf{o}\,\{s\} = q\,(\mathsf{o}'\,\{t\, s\})$,
3. $\forall\{s, p, p'\}.\, q\, p \oplus \{s\}\, q\, p' = q\,(p \oplus' \{t\, s\}\, p')$.

Here, again, Equations 2–3 are reminiscent of the laws of a monoid morphism, and specialize to them in the degenerate case $S = S' = 1$.

Similarly to containers, which form a category, directed containers form a category **DCont**.

The *interpretation* $[\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}}$ of a directed container is the set functor $[\![S \lhd P]\!]^{\mathrm{c}}$ together with natural transformations $\varepsilon : [\![S \lhd P]\!]^{\mathrm{c}} \to \mathsf{Id}$ and $\delta : [\![S \lhd P]\!]^{\mathrm{c}} \to$

$[\![S \lhd P]\!]^{\mathrm{c}} \cdot [\![S \lhd P]\!]^{\mathrm{c}}$, defined by

- $\varepsilon\,(s, v) = v\,(\mathsf{o}\,\{s\})$,
- $\delta\,(s, v) = (s, \lambda p.\,(s \downarrow p, \lambda p'.\,v\,(p \oplus \{s\}\,p')))$.

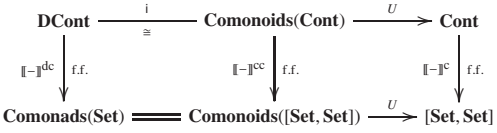
The natural transformations ε, δ satisfy the laws of a comonad. Therefore every directed container defines a comonad.

The obvious stream and non-empty list directed containers (with the suffixes structure) denote the expected comonads. The counit ε extracts the data at the root position (i.e., head of a stream or non-empty list) and the comultiplication δ replaces data at each position with the sub-datastructure rooted by that position. But if, instead, we consider for instance the sampling structure on the stream container, then the counit extracts the data at position 1 (the head of the tail) and the comultiplication sends a stream to a stream of its samplings.

The interpretation $[\![t \lhd q]\!]^{\mathrm{dc}}$ of a directed container morphism $t \lhd q$ is the natural transformation $[\![t \lhd q]\!]^{\mathrm{c}}$, which turns out to satisfy the laws of a comonad morphism.

We get that $[\![-]\!]^{\mathrm{dc}}$ is a functor from **DCont** to **Comonads**(**Set**) and that it is fully faithful.

Further, it turns out that directed containers correspond to exactly those containers whose interpretation carries a comonad structure. To be precise, **DCont** is isomorphic to **Comonoids**(**Cont**), and that in turn is easily seen to be the pullback of the forgetful functor U : **Comonads**(**Set**) $\to$ [**Set**, **Set**] along the fully faithful interpretation functor $[\![-]\!]^{\mathrm{c}}$: **Cont** $\to$ [**Set**, **Set**]:



More specifically, the comonoid corresponding to a directed container $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ under the isomorphism i is $(S \lhd P, t^{\varepsilon} \lhd q^{\varepsilon}, t^{\delta} \lhd q^{\delta})$ where $t^{\varepsilon}\ s = *$, $q^{\varepsilon}\,\{s\} * = \mathsf{o}\,\{s\}$, $t^{\delta}\ s = (s, \lambda p.\ s \downarrow p)$, $q^{\delta}\,\{s\}\,(p, p') = p \oplus p'$. The directed container for a comonoid $(S \lhd P, t^{\varepsilon} \lhd q^{\varepsilon}, t^{\delta} \lhd q^{\delta})$ is $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ where $s \downarrow p = \mathsf{snd}\,(t^{\delta}\ s)\,p$, $\mathsf{o}\,\{s\} = q^{\varepsilon}\,\{s\} *$ and $p \oplus p' = q^{\delta}\,(p, p')$. The equations of directed containers and their morphisms and those of comonoids on containers and their morphisms entail each other under this bijection. We see that, while this is not how we first arrived at it, the definition of directed containers can be derived systematically from instantiating the definition of comonoids for the category of containers and simplifying the result (in particular, there is no need for operations corresponding to t^{ε} and $\mathsf{fst} \circ t^{\delta}$ as it is forced that $\forall\{s\}.\, t^{\varepsilon}\ s = *$ and $\forall\{s\}.\, \mathsf{fst}\,(t^{\delta}\ s) = s$).

The functor $[\![-]\!]^{\mathrm{cc}}$: **Comonoids**(**Cont**) $\to$ **Comonoids**([**Set**, **Set**]) lifts the functor $[\![-]\!]^{\mathrm{c}}$: **Cont** $\to$ [**Set**, **Set**] in the expected way: $[\![S \lhd P, t^{\varepsilon} \lhd q^{\varepsilon}, t^{\delta} \lhd q^{\delta}]\!]^{\mathrm{cc}} = ([\![S \lhd P]\!]^{\mathrm{c}}, \varepsilon, \delta)$ where $\varepsilon = \mathsf{e}^{-1} \circ [\![t^{\varepsilon} \lhd q^{\varepsilon}]\!]^{\mathrm{c}}$ and $\delta = \mathsf{m}^{-1} \circ [\![t^{\delta} \lhd q^{\delta}]\!]^{\mathrm{c}}$.

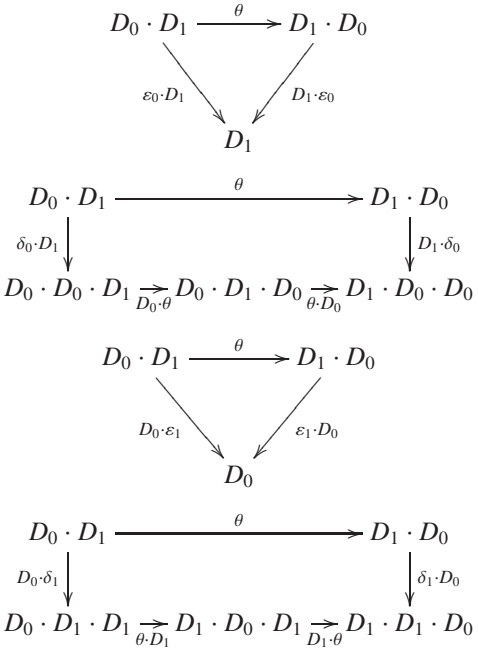
4 Distributive laws

We now proceed with our contribution, distributive laws and how they can be used to compose directed containers.

The composition of the underlying functors of two comonads need not have a comonad structure. Similarly, the composition of the underlying containers of two directed containers does not necessarily exhibit the structure of a directed container.

A sufficient condition for the composition of the underlying functors of two comonads to carry a comonad structure is that they distribute over each other (moreover, if we insist on a comonad structure compatible with the given ones, it is also necessary). It is natural to ask for a direct description of the corresponding condition for directed containers.

Recall first the definition of a distributive law between two comonads. For two comonads $(D_0, \varepsilon_0, \delta_0)$ and $(D_1, \varepsilon_1, \delta_1)$, a *distributive law* between them is a natural transformation $\theta : D_0 \cdot D_1 \to D_1 \cdot D_0$ making the following diagrams commute.



For directed containers, we seek a definition agreeing with this. First of all, a distributive law between two directed containers should determine one between their interpreting comonads. But since the interpreta-

tion of containers is fully-faithful, it makes also sense to aim at a bijection between the distributive laws between the two directed containers and the distributive laws between the two comonads.

The right definition is derived from fully-faithfulness of the interpretation of containers together with monoidality. Here is a sketch; the details are in Appendix A.

Given two directed containers $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$ and a container morphism $t^\theta \lhd q^\theta : (S_0 \lhd P_0) \cdot^{\mathrm{c}} (S_1 \lhd P_1) \to (S_1 \lhd P_1) \cdot^{\mathrm{c}} (S_0 \lhd P_0)$, the four distributive law equations for the natural transformation $\theta : [\![S_0 \lhd P_0]\!]^{\mathrm{c}} \cdot [\![S_1 \lhd P_1]\!]^{\mathrm{c}} \to [\![S_1 \lhd P_1]\!]^{\mathrm{c}} \cdot [\![S_0 \lhd P_0]\!]^{\mathrm{c}}$ given by $\theta = \mathsf{m}^{-1} \circ [\![t^\theta \lhd q^\theta]\!]^{\mathrm{c}} \circ \mathsf{m}$ translate into four equations about $t^\theta \lhd q^\theta$.

With abbreviations $(t_0\, s\, v, t_1\, s\, v) = t^\theta\, (s, v)$ and $(q_0\, \{s\}\, \{v\}\, p_1\, p_0, q_1\, \{s\}\, \{v\}\, p_1\, p_0) = q^\theta\, \{s, v\}\, (p_1, p_0)$, these four equations are split into 16 equations about t_0, t_1, q_0, q_1. One of them defines t_0 by $t_0\, s\, v = v\, (\mathsf{o}_0\, \{s\})$ and four are redundant.

Accordingly, we define a *distributive law* between two directed containers $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$ to be given by operations

- $t_1 : \Pi s : S_0.\, \Pi v : P_0\, s \to S_1.$
  $P_1\, (v\, (\mathsf{o}_0\, \{s\})) \to S_0,$
- $q_0 : \Pi \{s : S_0\}.\, \Pi \{v : P_0\, s \to S_1\}.$
  $\Pi p_1 : P_1\, (v\, (\mathsf{o}_0\, \{s\}))\, .\, P_0\, (t_1\, s\, v\, p_1)) \to P_0\, s,$
- $q_1 : \Pi \{s : S_0\}.\, \Pi \{v : P_0\, s \to S_1\}.$
  $\Pi p_1 : P_1\, (v\, (\mathsf{o}_0\, \{s\}))\, .\, \Pi p_0 : P_0\, (t_1\, s\, v\, p_1).$
  $P_1\, (v\, (q_0\, \{s\}\, \{v\}\, p_1\, p_0))$

satisfying the equations

1. $\forall \{s, v, p_1, p_0\}.$
   $t_1\, s\, v\, p_1 \downarrow_0 p_0 = t_1\, (s \downarrow_0 q_0\, p_1\, p_0)$
   $(\lambda p_0'.\, v\, (q_0\, p_1\, p_0 \oplus_0 p_0'))\, (q_1\, p_1\, p_0),$
2. $\forall \{s, v\}.\, t_1\, s\, v\, \mathsf{o}_1 = s,$
3. $\forall \{s, v, p_1, p_1'\}.$
   $t_1\, s\, v\, (p_1 \oplus_1 p_1') = t_1\, (t_1\, s\, v\, p_1)$
   $(\lambda p_0.\, v\, (q_0\, p_1\, p_0) \downarrow_1 q_1\, p_1\, p_0)\, p_1',$
4. $\forall \{s, v, p_1\}.\, q_0\, \{s\}\, \{v\}\, p_1\, \mathsf{o}_0 = \mathsf{o}_0\ ,$
5. $\forall \{s, v, p_1, p_0, p_0'\}.\, q_0\, \{s\}\, \{v\}\, p_1\, (p_0 \oplus_0 p_0') =$
   $q_0\, p_1\, p_0 \oplus_0 q_0\, (q_1\, p_1\, p_0)\, p_0',$
6. $\forall \{s, v, p_0\}.\, q_0\, \{s\}\, \{v\}\, \mathsf{o}_1\, p_0 = p_0,$
7. $\forall \{s, v, p_1, p_1', p_0\}.\, q_0\, \{s\}\, \{v\}\, (p_1 \oplus_1 p_1')\, p_0 =$
   $q_0\, p_1\, (q_0\, p_1'\, p_0),$
8. $\forall \{s, v, p_1\}.\, q_1\, \{s\}\, \{v\}\, p_1\, \mathsf{o}_0 = p_1,$
9. $\forall \{s, v, p_1, p_0, p_0'\}.\, q_1\, \{s\}\, \{v\}\, p_1\, (p_0 \oplus_0 p_0') =$
   $q_1\, (q_1\, p_1\, p_0)\, p_0',$
10. $\forall \{s, v, p_0\}.\, q_1\, \{s\}\, \{v\}\, \mathsf{o}_1\, p_0 = \mathsf{o}_1,$
11. $\forall \{s, v, p_1, p_1', p_0\}.\, q_1\, \{s\}\, \{v\}\, (p_1 \oplus_1 p_1')\, p_0 =$
    $q_1\, p_1\, (q_0\, p_1'\, p_0) \oplus_1 q_1\, p_1'\, p_0.$

With this definition, the distributive laws of two directed containers are in a bijection with the distributive laws of their interpreting comonads.

We saw before that directed containers generalize monoids and directed container morphisms generalize monoid morphisms. Against this background, we might expect that distributive laws of directed containers generalize some known construction for monoids. As it turns out, they do indeed. Namely, equations 4–11 governing the interaction of q_0, q_1 with o_0, $\oplus_0$, o_1, $\oplus_1$ generalize the laws of what is sometimes called a *matching pair*: a pair of mutual actions of two monoids on each other (Equations 6–7 and 8–9) satisfying some additional equations (Equations 4–5 and 10–11). In the special case $S_0 = S_1 = 1$, when the two directed containers degenerate to monoids, these equations specialize exactly to the equations of a matching pair.

A matching pair for two monoids equips the direct product of their carriers with a monoid structure compatible (in a certain sense) with the two given monoid structures. This monoid structure is called the *Zappa-Szép product* of these two monoids (or knit product, general product, bicrossed product, bilateral semidirect product). For an introduction into Zappa-Szép products of groups and monoids, we recommend the excellent paper by Brin [7]. We should now expect that distributive-law based composition of directed containers generalizes the Zappa-Szép product of monoids. In the next section, we will see that this is indeed the case.

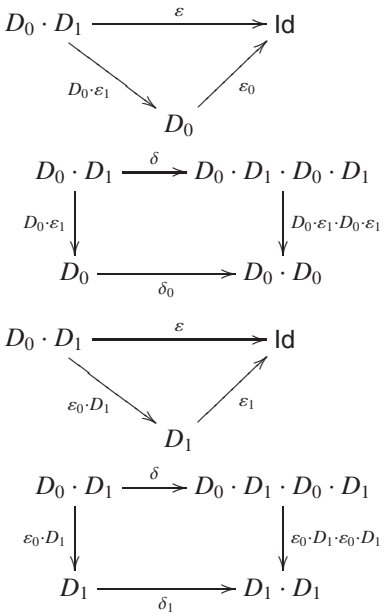
5 Composition of directed containers

We will now show how a distributive law between two directed containers can be used to compose them.

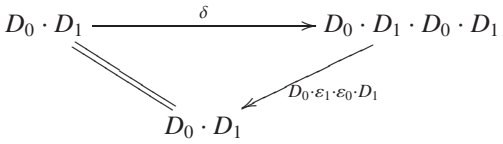
Recall that a distributive law θ between comonads $(D_0, \varepsilon_0, \delta_0)$ and $(D_1, \varepsilon_1, \delta_1)$ endows the functor $D = D_0 \cdot D_1$ with a comonad structure defined by

- $\varepsilon = \ D_0 \cdot D_1 \xrightarrow{\varepsilon_0 \cdot \varepsilon_1} \mathsf{Id}\ ,$
- $\delta = \ D_0 \cdot D_1 \xrightarrow{\delta_0 \cdot \delta_1} D_0 \cdot D_1 \cdot D_0 \cdot D_1$
  $\xrightarrow{D_0 \cdot \theta \cdot D_1} D_0 \cdot D_0 \cdot D_1 \cdot D_1\ .$

This given comonad structure is *compatible* with the given comonad structures on D_0, D_1 in the following sense. First, the natural transformations $\pi_0 : D \to D_0$, $\pi_1 : D \to D_1$, defined by $\pi_0 = D_0 \cdot \varepsilon_1$, $\pi_1 = \varepsilon_0 \cdot D_1$, turn out to be comonad morphisms, so $((D, \varepsilon, \delta), \pi_0, \pi_1)$ is a span on the two comonads. This means commutation of the following four diagrams (of whom the first and third are trivially equivalent to each other and to the equation $\varepsilon = \varepsilon_0 \cdot \varepsilon_1$).



In addition, the following condition (the *middle counital law*) is met.



Since a container whose interpretation carries a comonad structure is endowed with a directed container structure, it is easy to see that the composition of underlying containers of two directed containers with a distributive law between them must exhibit the structure of a directed container. Moreover, the induced directed container structure must be compatible with the two given directed container structures. But of course we aim at a direct construction of the composite directed container and a direct characterization of compatibility.

Given a distributive law (t_1, q_0, q_1) between two directed containers $(S_0 \lhd P_0, \mathsf{o}_0, \downarrow_0, \oplus_0)$ and $(S_1 \lhd P_1, \mathsf{o}_1, \downarrow_1, \oplus_1)$, we know that the composition $S \lhd P = (S_0 \lhd P_0) \cdot^{\mathrm{c}} (S_1 \lhd P_1)$ of the underlying containers is defined by

- $S = \Sigma s_0 : S_0.\, P_0\, s_0 \to S_1$,
- $P\,(s_0, v) = \Sigma p_0 : P_0\, s_0.\, P_1\,(v\, p_0)$,

The distributive law θ between the comonads $[\![S_0 \lhd P_0, \mathsf{o}_0, \downarrow_0, \oplus_0]\!]^{\mathrm{dc}}$ and $[\![S_1 \lhd P_1, \mathsf{o}_1, \downarrow_1, \oplus_1]\!]^{\mathrm{dc}}$ determines a comonad structure on the functor $[\![S_0 \lhd P_0]\!]^{\mathrm{c}} \cdot [\![S_1 \lhd P_1]\!]^{\mathrm{c}}$, which is isomorphic to $[\![S \lhd P]\!]^{\mathrm{c}}$. This comonad structure translates into the following directed container structure on the composite container $S \lhd P$:

- $(s_0, v) \downarrow (p_0, p_1) =$
  $(t_1\,(s_0 \downarrow_0 p_0)\,(\lambda p_0'.\, v\,(p_0 \oplus_0 p_0'))\, p_1,$
  $\lambda p_0'.\, v\,(p_0 \oplus_0 q_0\, p_1\, p_0') \downarrow_1 (q_1\, p_1\, p_0'))$,
- $\mathsf{o}\,\{s_0, v\} = (\mathsf{o}_0\,\{s_0\}, \mathsf{o}_1\,\{v\,(\mathsf{o}_0\,\{s_0\})\})$,
- $(p_0, p_1) \oplus (p_0', p_1') =$
  $(p_0 \oplus_0 q_0\, p_1\, p_0', q_1\, p_1\, p_0' \oplus_1 p_1')$.

The natural transformations $\pi_0 : [\![S_0 \lhd P_0]\!]^{\mathrm{c}} \cdot [\![S_1 \lhd P_1]\!]^{\mathrm{c}} \to [\![S_0 \lhd P_0]\!]^{\mathrm{c}}$, $\pi_1 : [\![S_0 \lhd P_0]\!]^{\mathrm{c}} \cdot [\![S_1 \lhd P_1]\!]^{\mathrm{c}} \to [\![S_1 \lhd P_1]\!]^{\mathrm{c}}$ translate to container morphisms $t^{\pi_0} \lhd q^{\pi_0} : S \lhd P \to S_0 \lhd P_0$ and $t^{\pi_1} \lhd q^{\pi_1} : S \lhd P \to S_1 \lhd P_1$, defined by

- $t^{\pi_0}\,(s_0, v) = s_0$,
- $q^{\pi_0}\,\{s_0, v\}\, p_0 = (p_0, \mathsf{o}_1\,\{v\, p_0\})$,
- $t^{\pi_1}\,(s_0, v) = v\,(\mathsf{o}_0\,\{s_0\})$,
- $q^{\pi_1}\,\{s_0, v\}\, p_1 = (\mathsf{o}_0\,\{s_0\}, p_1)$.

Since π_0, π_1 satisfy the conditions of a comonad morphism, $t^{\pi_0} \lhd q^{\pi_0}$ and $t^{\pi_1} \lhd q^{\pi_1}$ are directed container morphisms. This means that the following equations hold.

1. $\forall\{s_0, v, p_0\}$.
   $\mathsf{fst}\,((s_0, v) \downarrow (p_0, \mathsf{o}_1)) = s_0 \downarrow_0 p_0$,
2. $\forall\{s_0, v, p_1\}$.
   $\mathsf{snd}\,((s_0, v) \downarrow (\mathsf{o}_0, p_1))\, \mathsf{o}_0 = v\, \mathsf{o}_0 \downarrow_1 p_1$,
3. $\forall\{s_0, v\}.\, \mathsf{o}\,\{s_0, v\} = (\mathsf{o}_0, \mathsf{o}_1)$,
4. $\forall\{s_0, v, p_0, p_0'\}$.
   $(p_0, \mathsf{o}_1) \oplus \{s_0, v\}\,(p_0', \mathsf{o}_1) = (p_0 \oplus_0 p_0', \mathsf{o}_1)$,
5. $\forall\{s_0, v, p_1, p_1'\}$.
   $(\mathsf{o}_0, p_1) \oplus \{s_0, v\}\,(\mathsf{o}_0, p_1') = (\mathsf{o}_0, p_1 \oplus_1 p_1')$.

The middle counital law contributes two additional equations:

6. $\forall\{s_0, v, p_0\}$.
   $\mathsf{snd}\,((s_0, v) \downarrow (p_0, \mathsf{o}_1))\, \mathsf{o}_0 = v\, p_0$,
7. $\forall\{s_0, v, p_0, p_1\}$.
   $(p_0, \mathsf{o}_1) \oplus \{s_0, v\}\,(\mathsf{o}_0, p_1) = (p_0, p_1)$.

As we should expect, the definitions of o, $\oplus$ in terms of q_0, q_1 and Equations 3–5, 7 governing o, $\oplus$ remind of the construction of a Zappa-Szép product of two monoids from a matching pair resp. the general definition a Zappa-Szép product. We have obtained a generalization from the special case $S_0 = S_1 = 1$ to directed containers with multiple shapes.

For any two directed containers, their compatible compositions and the compatible compositions of the corresponding comonads are in a bijective correspondence.

For comonads, it is known that any compatible composite comonad arises from a distributive law; in fact, the distributive laws and compatible compositions of two comonads are in a bijective correspondence. If a comonad $(D_0 \cdot D_1, \varepsilon, \delta)$ is compatible with comonads $(D_0, \varepsilon_0, \delta_0)$ and $(D_1, \varepsilon_1, \delta_1)$, then the distributive law is

- $\theta = D_0 \cdot D_1 \xrightarrow{\delta} D_0 \cdot D_1 \cdot D_0 \cdot D_1 \xrightarrow{\varepsilon_0 \cdot D_1 \cdot D_0 \cdot \varepsilon_1} D_1 \cdot D_0$

For a directed container $((S_0 \triangleleft P_0) \cdot^{\mathrm{c}} (S_1 \triangleleft P_1), \downarrow, \mathsf{o}, \oplus)$ compatible with directed containers $(S_0 \triangleleft P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \triangleleft P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, the corresponding distributive law is defined by

- $t_1\ s_0\ v\ p_1 = \mathsf{fst}\,((s_0, v) \downarrow (\mathsf{o}_0, p_1))$,
- $q_0\ p_1\ p_0 = \mathsf{fst}\,((\mathsf{o}_0, p_1) \oplus (p_0, \mathsf{o}_1))$,
- $q_1\ p_1\ p_0 = \mathsf{snd}\,((\mathsf{o}_0, p_1) \oplus (p_0, \mathsf{o}_1))$.

The definition of q_0, q_1 from $\oplus$ generalizes the construction of a matching pair from a Zappa-Szép product.

From the bijections between the distributive laws of directed containers and the interpreting comonads and the compatible compositions of directed containers and the interpreting comonads, and from the bijection of the distributive laws and compatible compositions of comonads, it is immediate that the distributive laws between two given directed containers and compatible compositions of these directed containers are in a bijective correspondence. It is also possible to give a direct proof; we show most of these calculations in Appendix B.

6 Examples

Let us illustrate the constructions of the previous sections with examples of distributive laws of comonads and directed containers and the resulting composite comonads and directed containers. We start from simple and well-known examples and then move on to examples that we could only construct when we realized the analogy of compositions of directed containers to Zappa-Szép products of monoids.

Distributing over the product-with-a-constant comonad

First we look at the well-known fact that any comonad $(D_0, \varepsilon_0, \delta_0)$ distributes over the product-with-a-constant comonad $(D_1, \varepsilon_1, \delta_1)$ given by $D_1\,X = X \times A$, $\varepsilon_1 = \mathsf{fst}$, $\delta_1 = \langle \mathsf{id}, \mathsf{snd} \rangle$ where A is a fixed set. The distributive law $\theta : \forall\{X\}.\, D_0\,(X \times A) \to D_0\,X \times A$ is defined by $\theta = \langle D_0\,\mathsf{fst}, \varepsilon_0 \circ D_0\,\mathsf{snd} \rangle$.

The product-with-a-constant comonad is represented by the directed container with A many shapes of one position each. The precise definition is given by $S_1 = A$, $P_1\ s_1 = 1$, $s_1 \downarrow_1 * = s_1$, $\mathsf{o}_1\,\{s_1\} = *$ and $* \oplus_1 * = *$.

The distributive law of any directed container $(S_0 \triangleleft P_0, \mathsf{o}_0, \downarrow_0, \oplus_0)$ over $(S_1 \triangleleft P_1, \mathsf{o}_1, \downarrow_1, \oplus_1)$ is defined by

- $t_1\ s_0\ v * = s_0$,
- $q_0 * p_0 = p_0$,
- $q_1 * p_0 = *$.

As a consequence, the composite directed container is defined by

- $S = \Sigma s_0 : S_0.\, P_0\ s_0 \to A$,
- $P\,(s_0, v) = \Sigma p_0 : P_0\ s_0.\, 1$,
- $(s_0, v) \downarrow (p_0, *) = (s_0 \downarrow_0 p_0, \lambda p.\, v\,(p_0 \oplus_0 p))$,
- $\mathsf{o}\{s_0, v\} = (\mathsf{o}_0\,\{s_0\}, *)$,
- $(p_0, *) \oplus (p_0', *) = (p_0 \oplus_0 p_0', *)$.

Distributing the product-with-a-constant comonad

The product-with-a-constant comonad can also be composed with other comonads from the outside. For $(D_0, \varepsilon_0, \delta_0)$ the product-with-a-constant comonad given by a set A, the corresponding distributive law $\theta : \forall\{X\}.\, D_1 X \times A \to D_1\,(X \times A)$ is $\theta\,\{X\} = \sigma\,\{X, A\}$ where σ is the strength of D_1, defined (set-theoretically) by $\sigma\,(d, a) = D_1\,(\lambda x.\,(x, a))\,d$ (remember that any set functor is strong in a unique way).

Let $S_0 = A$, $P_0\ s_0 = 1$, $s_0 \downarrow_0 * = s_0$, $\mathsf{o}_0\,\{s_0\} = *$ and $* \oplus_0 * = *$. Then, for any directed container $(S_1 \triangleleft P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, we get a distributive law by setting

- $t_1\ s_0\ (\lambda *.\, s_1)\ p_1 = s_0$,
- $q_0\ p_1\ * = *$,
- $q_1\ p_1\ * = p_1$.

The composite directed container is defined by

- $S = \Sigma s_0 : A.\, 1 \to S_1$,
- $P\,(s_0, \lambda *.\, s_1) = \Sigma * : 1.\, P_1\ s_1$,
- $(s_0, \lambda *.\, s_1) \downarrow (*, p_1) = (s_0, \lambda *.\, s_1 \downarrow_1 p_1)$,
- $\mathsf{o}\,\{s_0, \lambda *.\, s_1\} = (*, \mathsf{o}_1\,\{s_1\})$,
- $(*, p_1) \oplus (*, p_1') = (*, p_1 \oplus_1 p_1')$.

Distributing a strict comonad

We can generalize the previous example by allowing $(D_0, \varepsilon_0, \delta_0)$ to be any strict comonad. By this we mean that $D_0\ X = X \times D_0^+ X$, $\varepsilon_0\,\{X\} = \mathsf{fst}$ and $\delta_0\,\{X\} = \langle \mathsf{id}\,\{D_0\,X\}, \delta_0^+\{X\} \circ \mathsf{snd} \rangle$ for some functor D_0^+ and natural transformation $\delta_0^+ : D_0^+ \to D_0^+ \cdot D_0$ that satisfy $(D_0^+ \cdot \varepsilon_0) \circ \delta_0^+ = \mathsf{Id}$ and $(D_0^+ \cdot \delta_0) \circ \delta_0^+ = (\delta_0^+ \cdot D_0) \circ \delta_0^+$. The distributive law $\theta : \forall\{X\}.\, D_1 X \times D_0^+\,(D_1 X) \to D_1\,(X \times D_0^+ X)$ is $\theta\,\{X\} = D_1\,(X \times D_0^+(\varepsilon_1\,\{X\})) \circ \sigma\,\{X, D_0^+\,(D_1 X)\}$ where σ is the strength of D_1.

A directed container $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ is strict if $p_0 \neq \mathsf{o}_0$ implies $p_0 \oplus_0 p_0' \neq \mathsf{o}_0$. The distributive law over any directed container $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$ is

- $t_1\ s\ v\ p_1 = s$,
- $q_0\ p_1\ p_0 = p_0$,
- $q_1\ p_1\ p_0 = \mathsf{if}\ p_0 = \mathsf{o}_0\ \mathsf{then}\ p_1\ \mathsf{else}\ \mathsf{o}_1$.

As a consequence, the composite directed container is defined by

- $S = \Sigma s_0 : S_0.\ P_0\ s_0 \to S_1$,
- $P\ (s_0, v) = \Sigma p_0 : P_0\ s_0.\ P_1\ (v\ p_0)$,
- $(s_0, v) \downarrow (p_0, p_1) = (s_0 \downarrow_0 p_0,$
  $\lambda p_0'.\ \mathsf{if}\ p_0' = \mathsf{o}_0\ \mathsf{then}\ v\ p_0 \downarrow_1 p_1\ \mathsf{else}\ v\ (p_0 \oplus_0 p_0'))$,
- $\mathsf{o}\ \{s_0, v\} = (\mathsf{o}_0\ \{s_0\}, \mathsf{o}_1\ \{(v\ \{s_0\})\})$,
- $(p_0, p_1) \oplus (p_0', p_1') =$
  $(p_0 \oplus_0 p_0', \mathsf{if}\ p_0' = \mathsf{o}_0\ \mathsf{then}\ p_1 \oplus_1 p_1'\ \mathsf{else}\ p_1')$.

It might be of interest to note that this is the definition one can easily arrive at when trying to compose two directed containers naively and directly, without thinking of distributive laws. It makes sense to refrain from applying the operations $\downarrow_1, \oplus_1$ to an inner position when the outer position is not the root position o_0. But directed container laws 2 and 5 fail for this composition unless the strictness assumption is made.

Distributing over the exponent comonad

Any comonad $(D_0, \varepsilon_0, \delta_0)$ distributes over the exponent comonad $(D_1, \varepsilon_1, \delta_1)$ given by $D_1\ X = A \to X$, $\varepsilon_1\ f = f\ e$, $\delta_1\ f\ p_1\ p_1' = f\ (p_1 \bullet p_1')$ where $(A, e, \bullet)$ is some fixed monoid. The distributive law $\theta : \forall\{X\}.\ D_0\ (A \to X) \to A \to D_0\ X$ is defined in terms of the strength σ of D_0 by $\theta = \mathsf{curry}\ (D_0\ \mathsf{eval} \circ \sigma)$.

The directed container corresponding to the exponent comonad $(D_1, \varepsilon_1, \delta_1)$ has one shape; the carrier of the monoid is its set of positions and the identity and multiplication are the root position and subshape position translation operation: $S_1 = *$, $P_1\ * = A$, $*\ \downarrow_1\ p_1 = *$, $\mathsf{o}_1\ \{*\} = e$, $p_1 \oplus_1 p_1' = p_1 \bullet p_1'$.

The corresponding distributive law of any directed container $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ over this directed container is given by

- $t_1\ s_0\ (\lambda\_.\ *)\ p_1 = *$,
- $q_0\ p_1\ p_0 = p_0$,
- $q_1\ p_1\ p_0 = p_1$.

The composite directed container is given by

- $S = \Sigma s_0 : S_0.\ P_0\ s_0 \to 1$,
- $P\ (s_0, \lambda\_.\ *) = \Sigma\_ : P_0\ s_0.\ A$,
- $(s_0, \lambda\_.\ *) \downarrow (p_0, p_1) = (s_0 \downarrow_0 p_0, \lambda\_.\ *)$,
- $\mathsf{o}\ \{s_0, \lambda\_.\ *\} = (\mathsf{o}_0\ \{s_0\}, e)$,
- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 \oplus_0 p_0', p_1 \bullet p_1')$.

If S_0 were also a singleton, the two given directed containers would both degenerate to monoids, q_0 and q_1 would be projections of the direct product of the carriers and the composite directed container would be the direct product of the two monoids.

So, e.g., in the special case of distributing the streams comonad over itself, one has $S_0 = S_1 = 1$, $P_0\ * = P_1\ * = \mathsf{Nat}$, $*\ \downarrow_0\ p = *\ \downarrow_1\ p = *$, $\mathsf{o}_0\ \{*\} = \mathsf{o}_1\ \{*\} = 0$, $p \oplus_0 p' = p \oplus_1 p' = p + p'$.

The distributive law between these directed containers is

- $t_1\ *\ v\ p_1 = *$,
- $q_0\ p_1\ p_0 = p_0$,
- $q_1\ p_1\ p_0 = p_1$.

The composite directed container is given by

- $S = \Sigma * : 1.\ \mathsf{Nat} \to 1$,
- $P\ (*, \lambda\_.\ *) = \Sigma\_ : \mathsf{Nat}.\ \mathsf{Nat}$,
- $(*, \lambda\_.\ *) \downarrow (p_0, p_1) = (*, \lambda\_.\ *)$,
- $\mathsf{o}\ \{*, \lambda\_.\ *\} = (0, 0)$,
- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 + p_0', p_1 + p_1')$.

Similarly we could distribute the non-empty lists comonad over the streams comonad.

Distributing the non-empty lists comonad over itself

But what about distributing the non-empty lists comonad over itself? Here the operation t_1 must be defined carefully to fulfill the laws. Intuitively, we need to ensure that the derived operation $\downarrow$ will behave well. A shape of a non-empty list of non-empty lists is a "skyline" (see Figure 1) specified by a natural number s_0 (its width) and an assignment v of natural numbers to any number $p_0 : [0, s_0]$ (its heights). A position in such a shape is a pair of natural numbers $p_0 : [0, s_0]$ and $p_1 : [0, v\ p_0]$ (the x- and y-coordinates). The width of the corresponding subshape can be at most $s_0 - p_0$, but we should also ensure that none of the heights is negative. A natural idea is to have the subshape as wide as possible within these constraints.

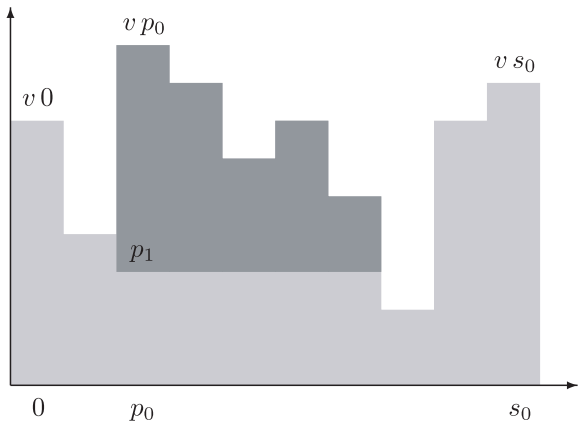


Fig. 1 A global shape and a subshape for non-empty lists of non-empty lists.

The two directed containers are given by $S_0 = S_1 = \mathsf{Nat}$, $P_0\,s = P_1\,s = [0, s]$, $s \downarrow_0 p = s \downarrow_1 p = s - p$, $\mathsf{o}_0\,\{s\} = \mathsf{o}_1\,\{s\} = 0$, $p \oplus_0 p' = p \oplus_1 p' = p + p'$.

The distributive law is specified by

- $t_1\,s_0\,v\,p_1 = \max\{s_0' : [0, s_0] \mid \forall p_0 : [0, s_0'].\ p_1 \le v\,p_0\}$,
- $q_0\,p_1\,p_0 = p_0$,
- $q_1\,p_1\,p_0 = p_1$.

The corresponding composite directed container is defined by

- $S = \Sigma s_0 : \mathsf{Nat}.\ [0, s_0] \to \mathsf{Nat}$,
- $P\,(s_0, v) = \Sigma p_0 : [0, s_0].\ [0, v\,p_0]$,
- $(s_0, v) \downarrow (p_0, p_1) = (\max\{s_0' : [0, s_0 - p_0] \mid \forall p_0' : [0, s_0'] \mid p_1 \le v\,(p_0 + p_0')\},\ \lambda p_0'.\ v\,(p_0 + p_0') - p_1)$,
- $\mathsf{o}\,\{s_0, v\} = (0, 0)$,
- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 + p_0', p_1 + p_1')$.

More ways to distribute the streams comonad over itself

Let us now return to the streams comonad. We are interested in finding other distributive laws of the streams comonad over itself than the one described above.

First of all it is immediate that we must have

- $t_1 * v\,p_1 = *$.

For the composite comonad we must therefore have

- $S = \Sigma * : 1.\ \mathsf{Nat} \to 1$,
- $P\,(*, \lambda\_.\ *) = \Sigma\_ : \mathsf{Nat}.\ \mathsf{Nat}$,
- $(*, \lambda\_.\ *) \downarrow (p_0, p_1) = (*, \lambda\_.\ *)$,
- $\mathsf{o}\,\{*, \lambda\_.\ *\} = (0, 0)$

Regarding q_0, q_1, the key is to realize that $(\mathsf{Nat}, 0, +)$ is the free monoid on one generator, 1. This allows us to use the method of Fernandes and Quinteiro [10] to generate the possible pairs of q_0 and q_1. The laws for q_0, q_1 can be rewritten as follows:

$$\begin{aligned}
q_0\,0\,p_0 &= p_0\\
q_0\,1\,0 &= 0\\
q_0\,1\,1 &= c_0\\
q_0\,1\,(2 + p_0) &= q_0\,1\,1 + q_0\,(q_1\,1\,1)\,(1 + p_0)\\
q_0\,(p_1 + 2)\,p_0 &= q_0\,(p_1 + 1)\,(q_0\,1\,p_0)
\end{aligned}$$

$$\begin{aligned}
q_1\,0\,p_0 &= 0\\
q_1\,1\,0 &= 1\\
q_1\,1\,1 &= c_1\\
q_1\,1\,(2 + p_0) &= q_1\,(q_1\,1\,1)\,(1 + p_0)\\
q_1\,(p_1 + 2)\,p_0 &= q_1\,(p_1 + 1)\,(q_0\,1\,p_0) + q_1\,1\,p_0
\end{aligned}$$

Here c_0, c_1 are some constants; they capture all the freedom that there is in choosing q_0, q_1. If $c_1 \le 1$, then the equations above make a structurally recursive definition of q_0, q_1. If $c_0 \le 1$, then symmetric equations form a structurally recursive definition.

Choosing $c_0 = c_1 = 1$ yields the solution we already described: q_0, q_1 are projections and $\oplus$ the multiplication operation of the direct product of the two monoids.

For $c_0 = c_1 = 0$, we get

- $q_0\,p_1\,p_0 = p_0 \dot{-} p_1$,
- $q_1\,p_1\,p_0 = p_1 \dot{-} p_0$.

where $\dot{-}$ denotes "truncated" subtraction, i.e., $p_0 \dot{-} p_1 = \mathsf{if}\ p_1 \le p_0\ \mathsf{then}\ p_0 - p_1\ \mathsf{else}\ 0$.

Accordingly, we have

- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 + (p_0' \dot{-} p_1), (p_1 \dot{-} p_0') + p_1')$.

For $c_0 = n + 1$, $c_1 = 0$, we get

- $q_0\,p_1\,p_0 = \mathsf{if}\ p_0 = 0\ \mathsf{then}\ 0\ \mathsf{else}\ n \times p_1 + p_0$,
- $q_1\,p_1\,p_0 = \mathsf{if}\ p_0 = 0\ \mathsf{then}\ p_1\ \mathsf{else}\ 0$,
- $(p_0, p_1) \oplus (p_0', p_1') = \mathsf{if}\ p_0' = 0\ \mathsf{then}\ (p_0, p_1 + p_1')\ \mathsf{else}\ (p_0 + n \times p_1 + p_0', p_1')$

whereas choosing $c_0 = n$, $c_1 = 1$ gives

- $q_0\,p_1\,p_0 = n^{p_1} \times p_0$,
- $q_1\,p_1\,p_0 = p_1$
- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 + n^{p_1} \times p_0', p_1 + p_1')$.

Distributing the streams and suffixes comonad over the streams and samplings comonad

We finish by looking at an example that involves two different monoid structures on the set of natural numbers. We want to distribute the streams and suffixes comonad over the streams and samplings comonad.

We let $S_0 = 1$, $P_0\, * = \mathsf{Nat}$, $* \downarrow_0 p = *$, $\mathsf{o}_0\,\{*\} = 0$, $p \oplus_0 p' = p + p'$ and $S_1 = 1$, $P_1\, * = \mathsf{Nat}$, $* \downarrow_1 p = *$, $\mathsf{o}_1\,\{*\} = 1$, $p \oplus_1 p' = p \times p'$.

A possible distributive law derives from the fact that $(\mathsf{Nat}, 1, \times)$ is (isomorphic to) a transformation monoid (namely, the full transformation monoid) of $(\mathsf{Nat}, 0, +)$ and thus acts on it. It is given by

- $t_1\, *\, (\lambda\_.\, *)\, p_1 = *$,
- $q_0\, p_1\, p_0 = p_1 \times p_0$,
- $q_1\, p_1\, p_0 = p_1$.

The composed directed container is therefore defined by

- $S = \Sigma * : 1.\, P_0\, * \to 1$,
- $P\,(*, \lambda\_.\, *) = \Sigma p_0 : P_0\, *.\, P_1\, *$,
- $(*, \lambda\_.\, *) \downarrow (p_0, p_1) = (*, \lambda\_.\, *)$,
- $\mathsf{o}\,\{*, \lambda\_.\, *\} = (0, 1)$,
- $(p_0, p_1) \oplus (p_0', p_1') = (p_0 + p_1 \times p_0', p_1 \times p_1')$.

7 Related work

Containers were introduced by Abbott, Altenkirch and Ghani [1]. Some generalizations are the indexed containers of Altenkirch and Morris [2] and quotient containers of Abbott et al. [4]. Simple/indexed containers are equivalent to the simple/dependent polynomial functors of Gambino and Hyland [11], now intensively studied by Kock [13], [14]. Gambino and Kock [12] have investigated polynomial monads.

Directed containers were first described in our recent work [3]. The idea is from the work of Uustalu and Vene [20] on the use of comonads for analyzing notions of context-dependence in computation. The most compelling examples of comonadic notions of context-dependence are dataflow computation, attribute evaluation and cellular automata [9], [18], [19]. Brookes and Geva [8] used comonads in a similar fashion for intensional semantics.

Distributive laws of monads and comonads are due to Beck [6]. They are well known in category theory and routinely used also in modern programming language semantics. Barr and Wells [5, Sec. 9.2] provide a concise overview of the most important facts about them (but axiomatize compatibility differently from Beck).

The Zappa-Szép product of groups and monoids was introduced by Zappa [17]. Recently, it has been studied and applied by Brin [7], Lawson [15] and several other authors.

8 Conclusions and future work

In this paper, we continued our research project on containers as a "syntax" for a class of datatypes with particularly good properties (sometimes informally referred to as strictly positive datatypes). We had previously introduced directed containers as a characterization of those containers whose interpretation carries the structure of a comonad. Here we turned our attention to composition of directed containers. We saw that, just as directed containers generalize monoids, distributive-law based composition of directed containers is a generalization of the Zappa-Szép product of monoids.

The Zappa-Szép product is relatively well known in algebra (more precisely, group theory and semigroup theory), but not at all so in theoretical computer science (mathematical structures for functional programming). We find such connections very exciting. One the one hand, we are presented with an opportunity to apply results from algebra (known facts about monoids) to computer science. In the course of this work, for instance, we learned about multiple examples of distributive laws between standard examples of comonads that we had not been aware of before and could only identify once we had realized that distributive laws are generalizations of mutual actions of monoids and acquainted ourselves with some results about these. On the other hand, we see new concepts (such as directed containers) emerge that have not been explored in algebra but appear natural in functional programming and promise to lead to elegant theory with useful applications that are also likely to interest algebraists.

We would like to continue this research by exploring further constructions on directed containers. We expect that we will witness more interaction between algebra and functional programming. The product of directed containers, for instance, must generalize the coproduct of monoids etc. There are many connections of semigroup theory to automata; we would like to find out whether there are interesting connections of directed containers to automata theory. It should be interesting to work out how comonads and constructions on comonads are characterized in alternative syntaxes for strictly positive datatypes, e.g., polynomial functors, and see where they come out prettiest. And of course we would like to see whether the various new facts we learn about datatypes with a comonad structure have applications to functional programming. In addition to comonads, one can also ask about monads. We have spelled out the data and laws for the correspond-

ing additional structure on containers [3], but have not studied them more closely.

We have formalized all of the directed container theory and examples from our FoSSaCS 2012 paper [3] in the dependently typed programming language Agda. This formalization is available online at http://cs.ioc.ee/~danel/dcont.html. We will grow this formalization to also cover distributive laws and composition of directed containers.

Acknowledgements

We would like to thank our referees for their diligent work.

This research was supported by the Estonian Ministry of Education and Research target-financed research theme no. 0140007s12, the Estonian Science Foundation grant no. 9475 and the Estonian Centre of Excellence in Computer Science, EXCS, a European Regional Development Fund funded project.

References

[1] M. Abbott, T. Altenkirch, and N. Ghani, "Containers: constructing strictly positive types," *Theor. Comput. Sci.*, vol. 342, no. 1, pp. 3–27, 2005.

[2] M. Abbott, T. Altenkirch, N. Ghani, and C. McBride, "Constructing polymorphic programs with quotient types," In D. Kozen, editor, *Proc. of 7th Int. Conf. on Mathematics of Program Construction, MPC 2004*, *Lect. Notes in Comput. Sci.*, vol. 3125, pp. 2–15, Springer, 2004.

[3] D. Ahman, J. Chapman, and T. Uustalu, "When is a container a comonad?", in L. Birkedal, editor, *Proc. of 15th Int. Conf. on Foundations of Software Science and Computation Structures, FoSSaCS 2012*, *Lect. Notes in Comput. Sci.*, vol. 7213, pp. 74–88, Springer, 2012.

[4] T. Altenkirch and P. Morris, "Indexed containers," in *Proc. of 24th Ann. IEEE Symp. on Logic in Computer Science, LICS 2009*, pp. 277–285, IEEE CS Press, 2009.

[5] M. Barr and C. Wells, *Toposes, Triples and Theories*, *Grundlehren der mathematischen Wissenschaften*, vol. 278, Springer, 1984.

[6] J. Beck, "Distributive laws," in B. Eckmann, ed., *Seminar on Triples and Categorical Homology, ETH 1966/67*, *Lect. Notes in Math.*, vol. 80, pp. 119–140, Springer, 1969.

[7] M. G. Brin, "On the Zappa-Szép product," *Commun. in Algebra*, vol. 33, no. 2, pp. 393–424, 2005.

[8] S. Brookes and S. Geva, "Computational comonads and intensional semantics," in M. P. Fourman, P. T. Johnstone, and A. M. Pitts, editors, *Applications of Categories in Computer Science*, *London Math. Society Lect. Note Series*, vol. 77, pp. 1–44. Cambridge Univ. Press, 1992.

[9] S. Capobianco and T. Uustalu, "A categorical outlook on cellular automata," in J. Kari, editor, *Proc. of 2nd Symp. on Cellular Automata, JAC 2010*, *TUCS Lecture Note Series*, vol. 13, pp. 88–89, Turku Centre for Comput. Sci., 2010.

[10] V. H. Fernandes and T. M. Quinteiro, "Bilateral semidirect product decompositions of transformation monoids," *Semigroup Forum*, vol. 82, no. 2, pp. 271–287, 2011.

[11] N. Gambino and M. Hyland, "Wellfounded trees and dependent polynomial functors," in S. Berardi, M. Coppo, and F. Damiani, editors, *Revised Selected Papers from Int. Wksh. on Types for Programs and Proofs, TYPES 2003*, *Lect. Notes in Comput. Sci.*, vol. 2085, pp. 210–225, Springer, 2004.

[12] N. Gambino and J. Kock, "Polynomial functors and polynomial monads," *Math. Proc. of Cambridge Phil. Soc.*, vol. 154, no. 1, pp. 153–192, 2013.

[13] J. Kock, *Notes on polynomial functors*, manuscript, 2009.

[14] J. Kock, "Polynomial functors and trees," *Int. Math. Research Notices*, vol. 2011, no. 3, pp. 609–673, 2011.

[15] M. V. Lawson, "A correspondence between a class of monoids and self-similar group actions I," *Semigroup Forum*, vol. 76, no. 3, pp. 489–517, 2008.

[16] U. Norell, "Towards a practical programming language based on dependent type theory," PhD thesis, Chalmers University of Technology, 2007.

[17] G. Zappa, "Sulla costruzione dei gruppi prodotto di due dati sottogruppi permutabili tra loro," in *Atti Secondo Congresso dell'Unione Matemática Italiana*, pp. 119–125. Edizioni Cremonense, Rome, 1942.

[18] T. Uustalu and V. Vene, "The essence of dataflow programming," in K. Yi, *Proc. of 2nd Asian Symp. on Programming Languages and Systems, APLAS 2004*, *Lect. Notes in Comput. Sci.*, vol. 3780, pp. 2–18, Springer, 2004.

[19] T. Uustalu and V. Vene, "Comonadic functional attribute evaluation," in M. van Eekelen, editor, *Trends in Functional Programming 6*, pp. 145–162, Intellect, Bristol, 2007.

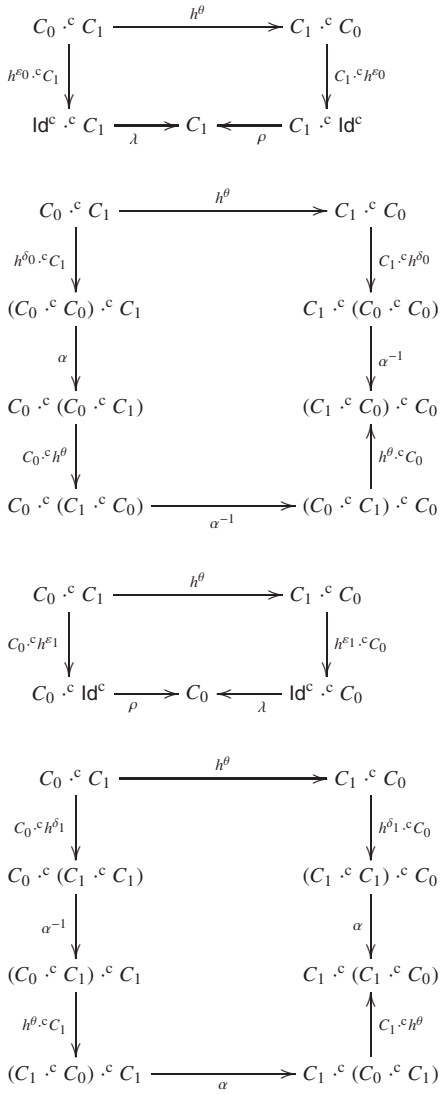
[20] T. Uustalu and V. Vene, "Comonadic notions of computation," in J. Adámek and C. Kupke, editors, *Proc. of 9th Int. Wksh. on Coalgebraic Methods in Computer Science, CMCS 2008*, *Electron. Notes in Theor. Comput. Sci.*, vol. 203, no. 5, pp. 263–284, Elsevier, 2008.

A Derivation of the definition of a distributive law between directed containers

We give the derivation of the definition of a distributive law between directed containers.

Given two comonoids in **Cont** $(C_0, h_0^\varepsilon, h_0^\delta) = (S_0 \lhd P_0, t^{\varepsilon_0} \lhd q^{\varepsilon_0}, t^{\delta_0} \lhd q^{\delta_0})$ and $(C_1, h_1^\varepsilon, h_1^\delta) =$

$(S_1 \lhd P_1, t^{\varepsilon_1} \lhd q^{\varepsilon_1}, t^{\delta_1} \lhd q^{\delta_1})$ and a container morphism $h^\theta = t^\theta \lhd q^\theta : C_0 \cdot^{\mathsf{c}} C_1 \to C_1 \cdot^{\mathsf{c}} C_0$, the natural transformation $\theta : [\![C_0]\!]^{\mathsf{c}} \cdot [\![C_1]\!]^{\mathsf{c}} \to [\![C_1]\!]^{\mathsf{c}} \cdot [\![C_0]\!]^{\mathsf{c}}$ defined by $\theta = \mathsf{m}^{-1} \circ [\![h^\theta]\!]^{\mathsf{c}} \circ \mathsf{m}$ satisfies the four equations of a distributive law between comonads if and only if h^θ satisfies the following four equations:



Let us abbreviate $(t_0\, s\, v, t_1\, s\, v) = t^\theta\,(s, v)$ and $(q_0\,\{s\}\,\{v\}\,p_1\,p_0, q_1\,\{s\}\,\{v\}\,p_1\,p_0) = q^\theta\,\{s, v\}\,(p_1, p_0)$. By expanding h^{ε_0}, h^{δ_0}, h^{ε_0}, h^{δ_1}, ρ, λ, α, these equations can be rewritten into the following 16 equations.

* $\forall\{s, v\}.\ t_0\, s\, v = v\,(\mathsf{o}_0\,\{s\})$,

$\forall\{s, v, p_1\}.\ q_0\,\{s\}\,\{v\}\,p_1\,(\mathsf{o}_0\,\{t_1\, s\, v\, p_1\}) = \mathsf{o}_0\,\{s\}$,

$\forall\{s, v, p_1\}.\ q_1\,\{s\}\,\{v\}\,p_1\,(\mathsf{o}_0\,\{t_1\, s\, v\, p_1\}) = p_1$,

* $\forall\{s, v\}.\ t_0\, s\, v =$
$t_0\, s\,(\lambda p_0.\ t_0\,(s \downarrow_0 p_0)\,(\lambda p_0'.\ v\,(p_0 \oplus_0 p_0')))$,

* $\forall\{s, v, p_1\}.\ t_1\, s\, v\, p_1 =$
$t_1\, s\,(\lambda p_0.\ t_0\,(s \downarrow_0 p_0)\,(\lambda p_0'.\ v\,(p_0 \oplus_0 p_0')))\, p_1$,

$\forall\{s, v, p_1, p_0\}.\ t_1\, s\, v\, p_1 \downarrow_0 p_0 =$
$t_1\,(s \downarrow_0 q_0\, p_1\, p_0)\,(\lambda p_0'.\ v\,(q_0\, p_1\, p_0 \oplus_0 p_0'))$
$(q_1\, p_1\, p_0)$,

$\forall\{s, v, p_1, p_0, p_0'\}.\ q_0\,\{s\}\,\{v\}\,p_1\,(p_0 \oplus_0 p_0') =$
$q_0\, p_1\, p_0 \oplus_0 q_0\,(q_1\, p_1\, p_0)\, p_0'$,

$\forall\{s, v, p_1, p_0, p_0'\}.\ q_1\,\{s\}\,\{v\}\,p_1\,(p_0 \oplus_0 p_0') =$
$q_1\,(q_1\, p_1\, p_0)\, p_0'$.

$\forall\{s, v\}.\ t_1\, s\, v\,(\mathsf{o}_1\,\{v\,(\mathsf{o}_0\,\{s\})\}) = s$,

$\forall\{s, v, p_0\}.\ q_0\,\{s\}\,\{v\}\,(\mathsf{o}_1\,\{v\,(\mathsf{o}_0\,\{s\})\})\, p_0 = p_0$,

$\forall\{s, v, p_0\}.\ q_1\,\{s\}\,\{v\}\,(\mathsf{o}_1\,\{v\,(\mathsf{o}_0\,\{s\})\})\, p_0 =$
$\mathsf{o}_1\,\{v\,(\mathsf{o}_0\,\{s\})\}$.

* $\forall\{s, v\}.\ t_0\, s\, v = t_0\, s\, v$,

* $\forall\{s, v, p_1\}.\ t_0\, s\, v \downarrow_1 p_1 =$
$t_0\,(t_1\, s\, v\, p_1)\,(\lambda p_0.\ v\,(q_0\, p_1\, p_0) \downarrow_1 q_1\, p_1\, p_0)$,

$\forall\{s, v, p_1, p_1'\}.\ t_1\, s\, v\,(p_1 \oplus_1 p_1') =$
$t_1\,(t_1\, s\, v\, p_1)\,(\lambda p_0.\ v\,(q_0\, p_1\, p_0) \downarrow_1 q_1\, p_1\, p_0)\, p_1'$,

$\forall\{s, v, p_1, p_1', p_0\}.\ q_0\,\{s\}\,\{v\}\,(p_1 \oplus_1 p_1')\, p_0 =$
$q_0\, p_1\,(q_0\, p_1'\, p_0)$,

$\forall\{s, v, p_1, p_1', p_0\}.\ q_1\,\{s\}\,\{v\}\,(p_1 \oplus_1 p_1')\, p_0 =$
$q_1\, p_1\,(q_0\, p_1'\, p_0) \oplus_1 q_1\, p_1'\, p_0$

Of the equations marked (*), the first one effectively defines t_0 and the remaining four become tautologies under that definition. The unmarked equations are exactly the equations 1–11 of a distributive law between directed containers. We see that we only need three operations t_1, q_0, q_1 in the definition of a distributive law between directed containers together with the 11 equations governing their interaction with $\downarrow_0$, o_0, $\oplus_0$, $\downarrow_1$, o_1, $\oplus_1$.

B Distributive laws vs. compatible composite directed containers

Given a distributive law between two directed containers, we get a compatible composite directed container.

Proof. Proof of directed container equation 1.

$$
\begin{aligned}
&(s_0, v) \downarrow \mathsf{o} \\
=\ & \quad \{\text{def. of } \mathsf{o}\} \\
&(s_0, v) \downarrow (\mathsf{o}_0, \mathsf{o}_1) \\
=\ & \quad \{\text{def. of } \downarrow\} \\
&(t_1\ (s_0 \downarrow_0 \mathsf{o}_0)\ (\lambda p_0'.\ v\ (\mathsf{o}_0 \oplus_0 p_0'))\ \mathsf{o}_1, \\
&\qquad \lambda p_0'.\ v\ (\mathsf{o}_0 \oplus_0 q_0\ \mathsf{o}_1\ p_0')) \downarrow_1 q_1\ \mathsf{o}_1\ p_0') \\
=\ & \quad \{\text{dir. cont. eqs. 1, 1, 4}\} \\
&(t_1\ s_0\ v\ \mathsf{o}_1, \lambda p_0'.\ v\ (q_0\ \mathsf{o}_1\ p_0')) \downarrow_1 q_1\ \mathsf{o}_1\ p_0') \\
=\ & \quad \{\text{dist. law eqs. 2, 6, 10}\} \\
&(s_0, \lambda p_0'.\ v\ p_0' \downarrow_1 \mathsf{o}_1) \\
=\ & \quad \{\text{dir. cont. eq. 1}\} \\
&(s_0, v)
\end{aligned}
$$

Proof of directed container equation 2.

$$
\begin{aligned}
&(s_0, v) \downarrow ((p_0, p_1) \oplus (p_0', p_1')) \\
=\ & \quad \{\text{def. of } \oplus\} \\
&(s_0, v) \downarrow (p_0 \oplus_0 q_0\ p_1\ p_0', q_1\ p_1\ p_0' \oplus_1 p_1') \\
=\ & \quad \{\text{def. of } \downarrow\} \\
&(t_1\ (s_0 \downarrow_0 (p_0 \oplus_0 q_0\ p_1\ p_0')) \\
&\qquad (\lambda p_0''.\ v\ ((p_0 \oplus_0 q_0\ p_1\ p_0') \oplus_0 p_0'')) \\
&\qquad (q_1\ p_1\ p_0' \oplus_1 p_1'), \\
&\quad \lambda p_0''.\ v\ ((p_0 \oplus_0 q_0\ p_1\ p_0') \oplus_0 \\
&\qquad q_0\ (q_1\ p_1\ p_0' \oplus_1 p_1')\ p_0'') \downarrow_1 \\
&\qquad\quad q_1\ (q_1\ p_1\ p_0' \oplus_1 p_1')\ p_0'') \\
=\ & \quad \{\text{dir. cont. eqs. 2, 5, 5}\} \\
&(t_1\ ((s_0 \downarrow_0 p_0) \downarrow_0 q_0\ p_1\ p_0') \\
&\qquad (\lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 p_0''))) \\
&\qquad (q_1\ p_1\ p_0' \oplus_1 p_1'), \\
&\quad \lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 \\
&\qquad q_0\ (q_1\ p_1\ p_0' \oplus_1 p_1')\ p_0'')) \downarrow_1 \\
&\qquad\quad q_1\ (q_1\ p_1\ p_0' \oplus_1 p_1')\ p_0'') \\
=\ & \quad \{\text{dist. law eqs. 3, 7, 11}\} \\
&(t_1\ (t_1\ ((s_0 \downarrow_0 p_0) \downarrow_0 q_0\ p_1\ p_0') \\
&\qquad\quad (\lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 p_0''))) \\
&\qquad\quad (q_1\ p_1\ p_0')) \\
&\qquad (\lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 \\
&\qquad\quad q_0\ (q_1\ p_1\ p_0')\ p_0'')) \downarrow_1 q_1\ (q_1\ p_1\ p_0')\ p_0'')\ p_1', \\
&\quad \lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 \\
&\qquad q_0\ (q_1\ p_1\ p_0')(q_0\ p_1'\ p_0''))) \downarrow_1 \\
&\qquad\quad (q_1\ (q_1\ p_1\ p_0')\ (q_0\ p_1'\ p_0'') \oplus_1 q_1\ p_1'\ p_0'')) \\
=\ & \quad \{\text{dist. law eqs. 5, 9, 5, 9}\} \\
&(t_1\ (t_1\ ((s_0 \downarrow_0 p_0)\ \downarrow_0 q_0\ p_1\ p_0') \\
&\qquad\quad (\lambda p_0''.\ v\ (p_0 \oplus_0 (q_0\ p_1\ p_0' \oplus_0 p_0''))) \\
&\qquad\quad (q_1\ p_1\ p_0')) \\
&\qquad\quad (\lambda p_0''.\ v\ (p_0 \oplus_0 q_0\ p_1\ (p_0' \oplus_0 p_0''))) \downarrow_1 \\
&\qquad\qquad q_1\ p_1\ (p_0' \oplus_0 p_0''))\ p_1', \\
&\qquad \lambda p_0''.\ v\ (p_0 \oplus_0 q_0\ p_1\ (p_0' \oplus_0 q_0\ p_1'\ p_0'')) \downarrow_1 \\
&\qquad\quad (q_1\ p_1\ (p_0' \oplus_0 q_0\ p_1'\ p_0'') \oplus_1 q_1\ p_1'\ p_0'')) \\
=\ & \quad \{\text{dist. law eq. 1, dir. cont. eq. 2}\} \\
&(t_1\ (t_1\ (s_0 \downarrow_0 p_0)\ (\lambda p_0''.\ v\ (p_0 \oplus_0 p_0''))\ p_1 \downarrow_0 p_0') \\
&\qquad\quad (\lambda p_0''.\ v\ (p_0 \oplus_0 q_0\ p_1\ (p_0' \oplus_0 p_0''))) \downarrow_1 \\
&\qquad\qquad q_1\ p_1\ (p_0' \oplus_0 p_0''))\ p_1', \\
&\qquad \lambda p_0''.\ (v\ (p_0 \oplus_0 q_0\ p_1\ (p_0' \oplus_0 q_0\ p_1'\ p_0''))) \downarrow_1 \\
&\qquad\quad q_1\ p_1\ (p_0' \oplus_0 q_0\ p_1'\ p_0'')) \downarrow_1 q_1\ p_1'\ p_0'') \\
=\ & \quad \{\text{def. of } \downarrow\} \\
&(t_1\ (s_0 \downarrow_0 p_0)\ (\lambda p_0''.\ v\ (p_0 \oplus_0 p_0''))\ p_1, \\
&\qquad \lambda p_0''.\ v\ (p_0 \oplus_0 q_0\ p_1\ p_0'') \downarrow_1 q_1\ p_1\ p_0'') \downarrow (p_0', p_1') \\
=\ & \quad \{\text{def. of } \downarrow\} \\
&((s_0, v) \downarrow (p_0, p_1)) \downarrow (p_0', p_1')
\end{aligned}
$$

Proof of directed container equation 3.

$$
\begin{aligned}
&(p_0, p_1) \oplus \mathsf{o} \\
=\ & \quad \{\text{def. of } \mathsf{o}\} \\
&(p_0, p_1) \oplus (\mathsf{o}_0, \mathsf{o}_1) \\
=\ & \quad \{\text{def. of } \oplus\} \\
&(p_0 \oplus_0 q_0\ p_1\ \mathsf{o}_0, q_1\ p_1\ \mathsf{o}_0 \oplus_1 \mathsf{o}_1) \\
=\ & \quad \{\text{dist. law eqs. 4, 8}\} \\
&(p_0 \oplus_0 \mathsf{o}_0, p_1 \oplus_1 \mathsf{o}_1) \\
=\ & \quad \{\text{dir. cont. eq. 3}\} \\
&(p_0, p_1)
\end{aligned}
$$

Proof of directed container equation 4.

$$
\begin{aligned}
&\mathsf{o} \oplus (p_0, p_1) \\
=\ & \quad \{\text{def. of } \mathsf{o}\} \\
&(\mathsf{o}_0, \mathsf{o}_1) \oplus (p_0, p_1) \\
=\ & \quad \{\text{def. of } \oplus\} \\
&(\mathsf{o}_0 \oplus_0 q_0\ \mathsf{o}_1\ p_0, q_1\ \mathsf{o}_1\ p_0 \oplus_1 p_1) \\
=\ & \quad \{\text{dist. law eqs. 6, 10}\} \\
&(\mathsf{o}_0 \oplus_0 p_0, \mathsf{o}_1 \oplus_1 p_1) \\
=\ & \quad \{\text{dir. cont. eq. 4}\} \\
&(p_0, p_1)
\end{aligned}
$$

Proof of directed container equation 5.

$$
\begin{aligned}
&((p_0, p_1) \oplus (p_0', p_1')) \oplus (p_0'', p_1'') \\
=\ & \quad \{\text{def. of } \oplus\} \\
&(p_0 \oplus_0 q_0\ p_1\ p_0', q_1\ p_1\ p_0' \oplus_1 p_1') \oplus (p_0'', p_1'') \\
=\ & \quad \{\text{def. of } \oplus\} \\
&((p_0 \oplus_0 q_0\ p_1\ p_0') \oplus_0 q_0\ (q_1\ p_1\ p_0' \oplus_1 p_1')\ p_0'', \\
&\quad q_1\ (q_1\ p_1\ p_0 \oplus_1 p_1')\ p_0'' \oplus_1 p_1'') \\
=\ & \quad \{\text{dir. cont. eq. 5, dist. law eq. 11}\}
\end{aligned}
$$

$$
\begin{aligned}
&(p_0 \oplus_0 (q_0\, p_1\, p_0' \oplus_0 q_0\, (q_1\, p_1\, p_0' \oplus_1 p_1')\, p_0''),\\
&\quad (q_1\, (q_1\, p_1\, p_0')\, (q_0\, p_1'\, p_0'') \oplus_1 q_1\, p_1'\, p_0'') \oplus_1 p_1'')\\
=\quad &\{\text{dist. law eq. 7, dist law eq. 9}\}\\
&(p_0 \oplus_0 (q_0\, p_1\, p_0' \oplus_0 q_0\, (q_1\, p_1\, p_0')\, (q_0\, p_1'\, p_0'')),\\
&\quad (q_1\, p_1\, (p_0' \oplus_0 q_0\, p_1'\, p_0'') \oplus_1 q_1\, p_1'\, p_0'') \oplus_1 p_1'')\\
=\quad &\{\text{dist. law eq. 5, dir. cont. eq. 5}\}\\
&(p_0 \oplus_0 q_0\, p_1\, (p_0' \oplus_0 q_0\, p_1'\, p_0''),\\
&\quad q_1\, p_1\, (p_0' \oplus_0 q_0\, p_1'\, p_0'') \oplus_1 (q_1\, p_1'\, p_0'' \oplus_1 p_1''))\\
=\quad &\{\text{def. of } \oplus\}\\
&(p_0, p_1) \oplus (p_0' \oplus_0 q_0\, p_1'\, p_0'', q_1\, p_1'\, p_0'' \oplus_1 p_1'')\\
=\quad &\{\text{def. of } \oplus\}\\
&(p_0, p_1) \oplus ((p_0', p_1') \oplus (p_0'', p_1''))
\end{aligned}
$$

Proof of compatibility equation 1.

$$
\begin{aligned}
&\mathsf{fst}\,((s_0, v) \downarrow (p_0, \mathsf{o}_1))\\
=\quad &\{\text{def. of } \downarrow\}\\
&t_1\,(s_0 \downarrow_0 p_0)\,(\lambda p_0'.\, v\,(p_0 \oplus_0 p_0'))\,\mathsf{o}_1\\
=\quad &\{\text{dist. law eq. 2}\}\\
&s_0 \downarrow_0 p_0
\end{aligned}
$$

Proof of compatibility equation 2.

$$
\begin{aligned}
&\mathsf{snd}\,((s_0, v) \downarrow (\mathsf{o}_0, p_1))\,\mathsf{o}_0\\
=\quad &\{\text{def. of } \downarrow\}\\
&v\,(\mathsf{o}_0 \oplus_0 q_0\, p_1\, \mathsf{o}_0) \downarrow_1 q_1\, p_1\, \mathsf{o}_0\\
=\quad &\{\text{dist. law eqs. 4, 8}\}\\
&v\,(\mathsf{o}_0 \oplus_0 \mathsf{o}_0) \downarrow_1 p_1\\
=\quad &\{\text{dir. cont. eqs. 3/4}\}\\
&v\,\mathsf{o}_0 \downarrow_1 p_1
\end{aligned}
$$

Proof of compatibility equation 3.

$$
\begin{aligned}
&\mathsf{o}\\
=\quad &\{\text{def. of } \mathsf{o}\}\\
&(\mathsf{o}_0, \mathsf{o}_1)
\end{aligned}
$$

Proof of compatibility equation 4.

$$
\begin{aligned}
&(p_0, \mathsf{o}_1) \oplus (p_0', \mathsf{o}_1)\\
=\quad &\{\text{def. of } \oplus\}\\
&(p_0 \oplus_0 q_0\, \mathsf{o}_1\, p_0', q_1\, \mathsf{o}_1\, p_0' \oplus_1 \mathsf{o}_1)\\
=\quad &\{\text{dist. law eqs. 6, 10}\}\\
&(p_0 \oplus_0 p_0', \mathsf{o}_1 \oplus_1 \mathsf{o}_1)\\
=\quad &\{\text{dir. cont. eq. 3/4}\}\\
&(p_0 \oplus_0 p_0', \mathsf{o}_1)
\end{aligned}
$$

Proof of compatibility equation 5.

$$
\begin{aligned}
&(\mathsf{o}_0, p_1) \oplus (\mathsf{o}_0, p_1')\\
=\quad &\{\text{def. of } \oplus\}\\
&(\mathsf{o}_0 \oplus_0 q_0\, p_1\, \mathsf{o}_0, q_1\, p_1\, \mathsf{o}_0 \oplus_1 p_1')\\
=\quad &\{\text{dist. law eqs. 4, 8}\}\\
&(\mathsf{o}_0 \oplus_0 \mathsf{o}_0, p_1 \oplus_1 p_1')\\
=\quad &\{\text{dir. cont. eq. 3/4}\}\\
&(\mathsf{o}_0, p_1 \oplus_1 p_1')
\end{aligned}
$$

Proof of compatibility equation 6.

$$
\begin{aligned}
&\mathsf{snd}\,((s_0, v) \downarrow (p_0, \mathsf{o}_1))\,\mathsf{o}_0\\
=\quad &\{\text{def. of } \downarrow\}\\
&v\,(p_0 \oplus_0 q_0\, \mathsf{o}_1\, \mathsf{o}_0) \downarrow_1 q_1\, \mathsf{o}_1\, \mathsf{o}_0\\
=\quad &\{\text{dist. law eqs. 4/6, 8/10}\}\\
&v\,(p_0 \oplus_0 \mathsf{o}_0) \downarrow_1 \mathsf{o}_1\\
=\quad &\{\text{dir. cont. eqs. 1, 3}\}\\
&v\,p_0
\end{aligned}
$$

Proof of compatibility equation 7.

$$
\begin{aligned}
&(p_0, \mathsf{o}_1) \oplus (\mathsf{o}_0, p_1)\\
=\quad &\{\text{def. of } \oplus\}\\
&(p_0 \oplus_0 q_0\, \mathsf{o}_1\, \mathsf{o}_0, q_1\, \mathsf{o}_1\, \mathsf{o}_0 \oplus_1 p_1)\\
=\quad &\{\text{dist. law eqs. 4/6, 8/10}\}\\
&(p_0 \oplus_0 \mathsf{o}_0, \mathsf{o}_1 \oplus_1 p_1)\\
=\quad &\{\text{dir. cont. eqs. 3, 4}\}\\
&(p_0, p_1)
\end{aligned}
$$

□

Given a compatible composite directed container of two directed containers, we get a distributive law.

Proof. Proof of distributive law equation 1.

$$
\begin{aligned}
&t_1\, s\, v\, p_1 \downarrow_0 p_0\\
=\quad &\{\text{def. of } t_1\}\\
&\mathsf{fst}\,((s, v) \downarrow (\mathsf{o}_0, p_1)) \downarrow_0 p_0\\
=\quad &\{\text{compat. eq. 1}\}\\
&\mathsf{fst}\,(((s, v) \downarrow (\mathsf{o}_0, p_1)) \downarrow (p_0, \mathsf{o}_1))\\
=\quad &\{\text{dir. cont. eq. 2}\}\\
&\mathsf{fst}\,((s, v) \downarrow ((\mathsf{o}_0, p_1) \oplus (p_0, \mathsf{o}_1)))\\
=\quad &\{\text{def. of } q_0, q_1\}\\
&\mathsf{fst}\,((s, v) \downarrow (q_0\, p_1\, p_0, q_1\, p_1\, p_0))\\
=\quad &\{\text{compat. eq. 7}\}\\
&\mathsf{fst}\,((s, v) \downarrow ((q_0\, p_1\, p_0, \mathsf{o}_1) \oplus (\mathsf{o}_0, q_1\, p_1\, p_0)))\\
=\quad &\{\text{dir. cont. eq. 2}\}\\
&\mathsf{fst}\,(((s, v) \downarrow (q_0\, p_1\, p_0, \mathsf{o}_1)) \downarrow (\mathsf{o}_0, q_1\, p_1\, p_0))\\
=\quad &\{\text{def. of } t_1\}\\
&t_1\,(\mathsf{fst}\,((s, v) \downarrow (q_0\, p_1\, p_0, \mathsf{o}_1)))\\
&\quad (\mathsf{snd}\,((s, v) \downarrow (q_0\, p_1\, p_0, \mathsf{o}_1)))\,(q_1\, p_1\, p_0)\\
=\quad &\{\text{compat. eq. 1}\}\\
&t_1\,(s \downarrow_0 q_0\, p_1\, p_0)\\
&\quad (\lambda p_0'.\, (\mathsf{snd}\,((s, v) \downarrow (q_0\, p_1\, p_0, \mathsf{o}_1)))\, p_0')\,(q_1\, p_1\, p_0)\\
=\quad &\{\text{compat. eq. 6}\}\\
&t_1\,(s \downarrow_0 q_0\, p_1\, p_0)
\end{aligned}
$$

$$
\begin{array}{ll}
, & (\lambda p_0'.\,\mathsf{snd}\,(((s,v)\downarrow(q_0\,p_1\,p_0,\mathsf{o}_1))\downarrow(p_0',\mathsf{o}_1))\,\mathsf{o}_0)\\
& (q_1\,p_1\,p_0)\\
= & \quad\{\text{dir. cont. eq. 2}\}\\
& t_1\,(s\downarrow_0 q_0\,p_1\,p_0)\\
& (\lambda p_0'.\,\mathsf{snd}\,((s,v)\downarrow((q_0\,p_1\,p_0,\mathsf{o}_1)\oplus(p_0',\mathsf{o}_1)))\,\mathsf{o}_0)\\
& (q_1\,p_1\,p_0)\\
= & \quad\{\text{compat. eq. 4}\}\\
& t_1\,(s\downarrow_0 q_0\,p_1\,p_0)\\
& (\lambda p_0'.\,\mathsf{snd}\,((s,v)\downarrow((q_0\,p_1\,p_0\oplus_0 p_0',\mathsf{o}_1)))\,\mathsf{o}_0)\\
& (q_1\,p_1\,p_0)\\
= & \quad\{\text{compat. eq. 6}\}\\
& t_1\,(s\downarrow_0 q_0\,p_1\,p_0)\,(\lambda p_0'.\,v\,(q_0\,p_1\,p_0\oplus_0 p_0'))\,(q_1\,p_1\,p_0)
\end{array}
$$

Proof of distributive law equation 2.

$$
\begin{array}{ll}
& t_1\,s\,v\,\mathsf{o}_1\\
= & \quad\{\text{def. of } t_1\}\\
& \mathsf{fst}\,((s,v)\downarrow(\mathsf{o}_0,\mathsf{o}_1))\\
= & \quad\{\text{compat. eq. 3}\}\\
& \mathsf{fst}\,((s,v)\downarrow\mathsf{o})\\
= & \quad\{\text{dir. cont. eq. 1}\}\\
& s
\end{array}
$$

Proof of distributive law equation 3.

$$
\begin{array}{ll}
& t_1\,s\,v\,(p_1\oplus_1 p_1')\\
= & \quad\{\text{def. of. } t_1\}\\
& \mathsf{fst}\,((s,v)\downarrow(\mathsf{o}_0,p_1\downarrow_1 p_1'))\\
= & \quad\{\text{compat. eq. 5}\}\\
& \mathsf{fst}\,((s,v)\downarrow((\mathsf{o}_0,p_1)\oplus(\mathsf{o}_0,p_1')))\\
= & \quad\{\text{dir. cont. eq. 2}\}\\
& \mathsf{fst}\,(((s,v)\downarrow(\mathsf{o}_0,p_1))\downarrow(\mathsf{o}_0,p_1'))\\
= & \quad\{\text{def. of } t_1\}\\
& t_1\,(\mathsf{fst}\,((s,v)\downarrow(\mathsf{o}_0,p_1)))\,(\mathsf{snd}\,((s,v)\downarrow(\mathsf{o}_0,p_1)))\,p_1'\\
= & \quad\{\text{def. of } t_1\}\\
& t_1\,(t_1\,s\,v\,p_1)\,(\mathsf{snd}\,((s,v)\downarrow(\mathsf{o}_0,p_1)))\,p_1'\\
= & \quad\{\text{compat. eq. 6}\}\\
& t_1\,(t_1\,s\,v\,p_1)\\
& (\lambda p_0.\,\mathsf{snd}\,(((s,v)\downarrow(\mathsf{o}_0,p_1))\downarrow(p_0,\mathsf{o}_1))\,\mathsf{o}_0)\,p_1'\\
= & \quad\{\text{dir. cont. eq. 2}\}\\
& t_1\,(t_1\,s\,v\,p_1)\\
& (\lambda p_0.\,\mathsf{snd}\,((s,v)\downarrow((\mathsf{o}_0,p_1)\oplus(p_0,\mathsf{o}_1)))\,\mathsf{o}_0)\,p_1'\\
= & \quad\{\text{def. of. } q_0,q_1\}\\
& t_1\,(t_1\,s\,v\,p_1)\\
& (\lambda p_0.\,\mathsf{snd}\,((s,v)\downarrow(q_0\,p_1\,p_0,q_1\,p_1\,p_0))\,\mathsf{o}_0)\,p_1'\\
= & \quad\{\text{compat. eq. 7}\}\\
& t_1\,(t_1\,s\,v\,p_1)
\end{array}
$$

Proof of distributive law equations 4 and 8.

$$
\begin{array}{ll}
& (q_0\,p_1\,\mathsf{o}_0,q_1\,p_1\,\mathsf{o}_0)\\
= & \quad\{\text{def. of. } q_0,q_1\}\\
& (\mathsf{o}_0,p_1)\oplus(\mathsf{o}_0,\mathsf{o}_1)\\
= & \quad\{\text{compat. eq. 3}\}\\
& (\mathsf{o}_0,p_1)\oplus\mathsf{o}\\
= & \quad\{\text{dir. cont. eq. 3}\}\\
& (\mathsf{o}_0,p_1)
\end{array}
$$

Proof of distributive law equations 5 and 9.

$$
\begin{array}{ll}
& (q_0\,p_1\,(p_0\oplus_0 p_0'),q_1\,p_1\,(p_0\oplus_0 p_0'))\\
= & \quad\{\text{def. of. } q_0,q_1\}\\
& (\mathsf{o}_0,p_1)\oplus(p_0\oplus_0 p_0',\mathsf{o}_1)\\
= & \quad\{\text{compat. eq. 4}\}\\
& (\mathsf{o}_0,p_1)\oplus((p_0,\mathsf{o}_1)\oplus(p_0',\mathsf{o}_1))\\
= & \quad\{\text{dir. cont. eq. 5}\}\\
& ((\mathsf{o}_0,p_1)\oplus(p_0,\mathsf{o}_1))\oplus(p_0',\mathsf{o}_1)\\
= & \quad\{\text{def. of. } q_0,q_1\}\\
& (q_0\,p_1\,p_0,q_1\,p_1\,p_0)\oplus(p_0',\mathsf{o}_1)\\
= & \quad\{\text{compat. eq. 7}\}\\
& ((q_0\,p_1\,p_0,\mathsf{o}_1)\oplus(\mathsf{o}_0,q_1\,p_1\,p_0))\oplus(p_0',\mathsf{o}_1)\\
= & \quad\{\text{dir. cont. eq. 5}\}\\
& (q_0\,p_1\,p_0,\mathsf{o}_1)\oplus((\mathsf{o}_0,q_1\,p_1\,p_0)\oplus(p_0',\mathsf{o}_1))\\
= & \quad\{\text{def. of. } q_0,q_1\}\\
& (q_0\,p_1\,p_0,\mathsf{o}_1)\oplus\\
& \quad(q_0\,(q_1\,p_1\,p_0)\,p_0',q_1\,(q_1\,p_1\,p_0)\,p_0')\\
= & \quad\{\text{compat. eq. 7}\}\\
& (q_0\,p_1\,p_0,\mathsf{o}_1)\oplus\\
& \quad((q_0\,(q_1\,p_1\,p_0)\,p_0',\mathsf{o}_1)\oplus(\mathsf{o}_0,q_1\,(q_1\,p_1\,p_0)\,p_0'))\\
= & \quad\{\text{dir. cont. eq. 5}\}\\
& ((q_0\,p_1\,p_0,\mathsf{o}_1)\oplus(q_0\,(q_1\,p_1\,p_0)\,p_0',\mathsf{o}_1))\oplus\\
& \quad(\mathsf{o}_0,q_1\,(q_1\,p_1\,p_0)\,p_0')
\end{array}
$$

$$
\begin{aligned}
&= \quad \{\text{compat. eq. 4}\}\\
&\quad (q_0\, p_1\, p_0 \oplus_0 q_0\, (q_1\, p_1\, p_0)\, p_0', \mathsf{o}_1) \oplus\\
&\qquad (\mathsf{o}_0, q_1\, (q_1\, p_1\, p_0)\, p_0')\\
&= \quad \{\text{compat. eq. 7}\}\\
&\quad (q_0\, p_1\, p_0 \oplus_0 q_0\, (q_1\, p_1\, p_0)\, p_0', q_1\, (q_1\, p_1\, p_0)\, p_0'))
\end{aligned}
$$

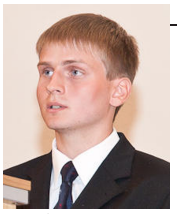
Proof of distributive law equations 6 and 10.

$$
\begin{aligned}
&(q_0\, \mathsf{o}_1\, p_0, q_1\, \mathsf{o}_1\, p_0)\\
&= \quad \{\text{def. of. } q_0, q_1\}\\
&(\mathsf{o}_0, \mathsf{o}_1) \oplus (p_0, \mathsf{o}_1)\\
&= \quad \{\text{compat. eq. 3}\}\\
&\mathsf{o} \oplus (p_0, \mathsf{o}_1)\\
&= \quad \{\text{dir. cont. eq. 4}\}\\
&(p_0, \mathsf{o}_1)
\end{aligned}
$$

Proof of distributive law equations 7 and 11.

$$
\begin{aligned}
&(q_0\, (p_1 \oplus_1 p_1')\, p_0, q_1\, (p_1 \oplus_1 p_1')\, p_0)\\
&= \quad \{\text{def. of. } q_0, q_1\}\\
&(\mathsf{o}_0, p_1 \oplus_1 p_1') \oplus (p_0, \mathsf{o}_1)\\
&= \quad \{\text{compat. eq. 5}\}\\
&((\mathsf{o}_0, p_1) \oplus (\mathsf{o}_0, p_1')) \oplus (p_0, \mathsf{o}_1)\\
&= \quad \{\text{dir. cont. eq. 5}\}\\
&(\mathsf{o}_0, p_1) \oplus ((\mathsf{o}_0, p_1') \oplus (p_0, \mathsf{o}_1))\\
&= \quad \{\text{def. of. } q_0, q_1\}\\
&(\mathsf{o}_0, p_1) \oplus (q_0\, p_1'\, p_0, q_1\, p_1'\, p_0)\\
&= \quad \{\text{compat. eq. 7}\}\\
&(\mathsf{o}_0, p_1) \oplus ((q_0\, p_1'\, p_0, \mathsf{o}_1) \oplus (\mathsf{o}_0, q_1\, p_1'\, p_0))\\
&= \quad \{\text{dir. cont. eq. 5}\}\\
&((\mathsf{o}_0, p_1) \oplus (q_0\, p_1'\, p_0, \mathsf{o}_1)) \oplus (\mathsf{o}_0, q_1\, p_1'\, p_0)\\
&= \quad \{\text{def. of. } q_0, q_1\}\\
&(q_0\, p_1\, (q_0\, p_1'\, p_0), q_1\, p_1\, (q_0\, p_1'\, p_0)) \oplus\\
&\qquad (\mathsf{o}_0, q_1\, p_1'\, p_0)\\
&= \quad \{\text{compat. eq. 7}\}\\
&((q_0\, p_1\, (q_0\, p_1'\, p_0), \mathsf{o}_1) \oplus (\mathsf{o}_0, q_1\, p_1\, (q_0\, p_1'\, p_0))) \oplus\\
&\qquad (\mathsf{o}_0, q_1\, p_1'\, p_0)\\
&= \quad \{\text{dir. cont. eq. 5}\}\\
&(q_0\, p_1\, (q_0\, p_1'\, p_0), \mathsf{o}_1) \oplus\\
&\qquad ((\mathsf{o}_0, q_1\, p_1\, (q_0\, p_1'\, p_0)) \oplus (\mathsf{o}_0, q_1\, p_1'\, p_0))\\
&= \quad \{\text{compat. eq. 5}\}\\
&(q_0\, p_1\, (q_0\, p_1'\, p_0), \mathsf{o}_1) \oplus\\
&\qquad (\mathsf{o}_0, q_1\, p_1\, (q_0\, p_1'\, p_0) \oplus_1 q_1\, p_1'\, p_0)\\
&= \quad \{\text{compat. eq. 7}\}\\
&(q_0\, p_1\, (q_0\, p_1'\, p_0), q_1\, p_1\, (q_0\, p_1'\, p_0) \oplus_1 q_1\, p_1'\, p_0)
\end{aligned}
$$

□



Danel AHMAN

Danel AHMAN (b. 1988) is a PhD student at the University of Edinburgh with a BSc (2010) from the Tallinn University of Technology and MPhil from the University of Cambridge (2012). His research interests are in category theory and programming language semantics.

Tarmo UUSTALU

Tarmo UUSTALU (b. 1969) is a leading researcher at the Institute of Cybernetics, the Tallinn University of Technology (TUT) and is currently leading the Estonian Centre of Excellence in Computer Science, EXCS. He has an MSc degree from TUT (1992) and PhD from the Royal Institute of Technology, Stockholm (1998). His research interests are in type theory, category theory, functional programming, programming language semantics, type systems and program logics.