An Introduction to Coalgebras

u7646615

May 2024

1 Introduction

This report serves as a short introduction into the theory of coalgebras for someone with a working knowledge of the basics of category theory. The formulation of coalgebras is a fundamentally category theoretic exercise, however most work with coalgebras extends out of category theory. Due to the nature of this course, this report aligns itself strongly with the category theoretic interpretation. We follow Rutten's introductory paper "Universal Coalgebra: a theory of systems"[9] as my main source. I extend from this paper to fill in details of the proofs Rutten provides, or to explore other interesting aspects of the theory of coalgebras. I generalise some proofs from just making sense in **Sets**.
I aim to build towards the coinduction proof and definition principles which form some of the fundamental theory of coalgebras.

2 F-Algebras

Definition 2.1. *The category* $\textbf{\textit{Algebra}}_{\mathcal{C}_F}$ *is induced by a category* $\mathcal{C}$ *and an endofunctor* $F : \mathcal{C} \to \mathcal{C}$. *It has objects as tuples* (A, α) *where* $A \in \mathcal{C}$ *and some arrow* $\alpha : F(A) \to A$ *from* $\mathcal{C}$. *It has arrows* $f : (A, \alpha) \to (B, \beta)$ *which are arrows of* $\mathcal{C}$ *such that:*

$$\begin{array}{ccc} F(A) & \xrightarrow{\alpha} & A \\ \downarrow{\scriptstyle Ff} & & \downarrow{\scriptstyle f} \\ F(B) & \xrightarrow{\beta} & B \end{array}$$

Commutes.

The objects of this category are called algebras and the arrows are algebra homomorphisms.

In terms of the history of F-algebras they initially arose due to their value in homological algebra [1]. It is clear where their name comes from as we can use this general encoding to define familiar algebras. If we take $F : \textbf{Sets} \to \textbf{Set}$ to be $F(G) = 1 + G + G \times G$ all groups will be in our Algebra category. However, not all objects of this category are groups. The functor F indicates that any algebra (A, α) of this category has a distinguished element by the restriction of α to 1, a unary operation by restriction to G and a binary operation by restriction to $G \times G$. Indeed all groups have this but our encoding says nothing of how α must behave or how these operations interact. It should be mentioned that in this category the algebras that are groups are precisely the algebras that have a monad structure, but this will be expanded on in section 8.

Suppose that (A, α) and (B, β) are both groups and we have some algebra homomorphism between them called f. We depict elements of $F(A)$ as (a, i) where $i = 0, 1$ or 2 depicting if a comes from 1, A or $A \times A$ respectively. Then $f \circ \alpha(a, i) = \beta \circ Ff(a, i)$. If $i = 0$ then

$$f \circ \alpha(*, 0) = f(u_A)$$
$$\beta \circ Ff(*, 0) = \beta(*, 0) = u_B$$

So $u_B = f(u_A)$.
It is easy to see that f also respects composition and thus f is a group homomorphism. This is desirable as we can use the general language of category theory to study universal algebras.

Such a perspective on algebra has lead to its use in computational semantics and type theory. Of note is [6] which in 1990 gave an in depth demonstration of using initial F-algebras to define recursive data-types in the lambda calculus. F-algebra has since been implemented into the Haskell package "Control.Functor.Algebra"and its connection to Wadler's work in [6] is highlighted by [14]. This is significant as this is where the applications of modern coalgebras that I am mostly interested in lie. These applications of algebras use the notion of an initial algebra, an initial object in $\mathbf{Algebra}_{\mathcal{C}_F}$. When a data-type is an initial coalgebra one can depict functions on such a data-type as algebra homomorphisms to another Algebra, which can be used as a tool to prove that such a function is a well defined computation. See [6] on how such a technique can be used to define inductive data-types.

3 Coalgebras

The value of a coalgebra in studying data-types is that modelling a data-type with a terminal coalgebra allows for what could be described as a behavior focused interpretation of data-types. Functions of data-types defined by terminal coalgebras need not terminate to be computable, of course assuming they are evaluated lazily. This is pivotal for studying data-types like streams which do not have a base case. I now move from focusing on motivation to introducing the theory of coalgebras.

Definition The category $\mathcal{C}_F$ is induced by a category $\mathcal{C}$ and an endofunctor $F : \mathcal{C} \to \mathcal{C}$. It has objects as tuples (S, α_S) where $S \in \mathcal{C}$ and $\alpha_S : S \to F(S)$. It has arrows $f : (S, \alpha_S) \to (T, \alpha_T)$ which are arrows coming from $\mathcal{C}$ such that:

$$\begin{tikzcd} F(S) \arrow[d, "Ff"'] & S \arrow[l, "\alpha_S"'] \arrow[d, "f"] \\ F(T) & T \arrow[l, "\alpha_T"'] \end{tikzcd}$$

commutes.
A coalgebra is an object in such a category, and a coalgebra homomorphism (or just homomorphism) is an arrow. The coalgebra (S, α_S) has type S and transition structure α_S. It is immediate from this definition the reason for the name coalgebra. A coalgebra is just the dual of algebra.
We must also verify that $\mathcal{C}_F$ is actually a category. First we see that the composition of homomorphisms is a homomorphism. Consider a homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$ and $g : (T, \alpha_T) \to (W, \alpha_W)$.

$$\begin{tikzcd} S \arrow[r, "f"] \arrow[d, "\alpha_S"] \arrow[rr, bend left, "g\circ f"] & T \arrow[r, "g"] \arrow[d, "\alpha_T"] & W \arrow[d, "\alpha_W"] \\ FS \arrow[r, dashed, "Ff"] \arrow[rr, bend right, "F(g\circ f)"'] & FT \arrow[r, dashed, "Fg"] & FW \end{tikzcd}$$

Then gf is a homomorphism $(S, \alpha_S) \to (W, \alpha_W)$ by:

$$\begin{aligned} \alpha_W \circ g \circ f &= (Fg) \circ \alpha_T \circ f \\ &= (Fg) \circ (Ff) \circ \alpha_S \\ &= F(g \circ f) \circ \alpha_S \end{aligned}$$

We see also that $id_{(S,S_\alpha)}$ is just id_S as id_S is indeed an arrow of $\mathcal{C}_F$ ($\alpha_S \circ id_S = id_{FS} \circ \alpha_S = F(id_S) \circ \alpha_S$) and clearly for any $f : (S, \alpha_S) \to A$, $f \circ id_S = f$ and likewise for any $g : A \to (S, \alpha_S)$, $id_S \circ g = g$. So indeed $\mathcal{C}_F$ is a category.

4 Systems

We would be remiss not to mention Rutten's choice to call coalgebras "systems". We take a system to be a mathematical object defined by the way behave and change, we are more interested in their dynamics then the elements that comprise them. A coalgebra precisely captures this, as we saw in the definition of a coalgebra, the type of the coalgebra does not meaningfully describe it.

We touch on the history of coalgebra, with the inception of a coalgebra as the dual of an algebra. Much like algebras, coalgebras have found relevance in homological algebra [1]. However, the majority of the field's development can be attributed to being an abstraction of a system. This dates back to at least 1982 [3] where Manes uses coalgebras as a means of describing the semantics of automata systems. Maybe a more fundamental example would be in 1988 [4] where Aczel innovated on Milner's calculus of communicating systems (CCS), a tool to study the dynamics of multiple connected systems, by viewing it as coalgebra. This is were the term transition structure of a coalgebra comes from.

With this perspective on coalgebras one is primarily interested in the category of sets. One may easily construct a category of coalgebras over any category. However, if we keep in mind the historical development of coalgebras in studying system dynamics and parameterized data-types any choice of base category other than **Sets** is generally unhelpful. That said see [2] for an enlightening use of coalgebras induced from the category of sets and relations, which seem to hold similar potential in applications as **Sets** has shown.

Consider some examples of coalgebras over sets. Our first example is if we take F to be the identity. This may seem trivial but our transition structures can still be interesting. Consider the Fibonacci sequence [13]:

$$\begin{aligned} &fib : \mathbb{N} \to \mathbb{N} \\ &fib(n) = \left\{ \begin{array}{cc} 0, & n = 0 \\ 1, & n = 1 \\ fib(n-1) + fib(n-2), & \text{else} \end{array} \right\} \end{aligned}$$

Then $(\mathbb{N}, fib)$ is a coalgebra of this type but could be hardly considered trivial. In general we write

$$\begin{aligned} &s \to s' \\ \iff &s' = \alpha_S(s) \end{aligned}$$

The above can be interpreted as s transitions to s'.

On Aczel's work with CCS consider F as $F(S) = A \times S$, computationally we think of A as the set of possible outputs. We write $s \underset{a}{\to} s'$ when $(a, s') = \alpha_S(s)$ saying that s transitions to s' emitting a. This models CCS exactly where every individual system changes its state $s \to s'$ and emits some signal a that may influence the behavior of other systems. More broadly we get a notion of observable and internal computation. We can model many elementary systems with basic operations of objects in **Sets**. If $F(S) = S^A$, A is now our input language. We write $s \xrightarrow{a} s'$ when $s' = \alpha_A(s)(a)$. We can define Mealy or Moore machines, which are deterministic automata with $F(S) = (B \times S)^A$ where the output signal is determined by the input signal or $B \times S^A$ where the output signal is not dependant on the input respectively. We write $s \xrightarrow[b]{a} s'$ if $(b, s') = \alpha_S(s)(a)$.
We may model termination of a process with $F(S) = A + S$. If $\alpha(s) \in A$ then by the fact that $\mathrm{Dom}(\alpha_S) = S$ there are no further transitions that can occur, we then write $\downarrow s$ if $\alpha(s) \in A$. We can can even study non-deterministic systems with the power set functor writing $s \to s'$ if $s' \in \alpha(s)$. Although, this does have the unfortunate effect of inducing a category without a terminal coalgebra so we may be inclined to use the finite power set functor which gives only finite subsets.

5 The Basics of Working with Coalgebras

Here I introduce some fundamental tools for studying coalgebras.

Lemma 5.1. *[9][1] (Trivially generalised) If $h = g \circ f$ in $\mathcal{C}$, but h and f are homomorphisms and f is epic then g is also a homomorphism.*

Proof.

$$\begin{tikzcd} S \arrow[r, "f"] \arrow[rr, bend left, "h"] \arrow[d, "\alpha_S"] & T \arrow[r, "g"] \arrow[d, "\alpha_T"] & W \arrow[d, "\alpha_W"] \\ FS \arrow[r, "Ff"] \arrow[rr, bend right, "Fh"'] & FT \arrow[r, "Fg"] & FW \end{tikzcd}$$

$$\begin{aligned} \alpha_w \circ h &= Fh \circ \alpha_S \\ \alpha_w \circ g \circ f &= Fg \circ Ff \circ \alpha_S \\ \alpha_w \circ g \circ f &= Fg \circ \alpha_T \circ f \\ \implies \alpha_w \circ g &= Fg \circ \alpha_T, \ [f \text{ is epic}] \end{aligned}$$

[1] I will mark theorems and lemmas with where I found them, and a note about any significant changes I may have made. If I have worked a theorem or lemma out myself it will be left unmarked.

This proof while unremarkable is a simple example of how many theorems about coalgebras are often proven.

Lemma 5.2. *[9] If $\phi \in \mathcal{C}_1$ is an isomorphism then ϕ is a homomorphism if and only if ϕ^{-1} is.*

Bisimulations are a type a binary relation between two transition systems. Aczel's work in 1989 [5] is at the very least the most well known generalization of bisimulations to the theory of coalgebras. We will see later that this is a very powerful tool.

Definition 5.3. *A bisimulation between (S, α_S) and (T, α_T) is an object $(R, \alpha_r : R \to FR)$ such that R is a subobject of $S \times T$ (in $\mathcal{C}$) and the projections of R form coalgebra homomorphisms. In diagram form*

$$\begin{tikzcd}
S \arrow[d, "\alpha_s"] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha_r"] & T \arrow[d, "\alpha_t"] \\
FS & FR \arrow[l, "F\pi_1"'] \arrow[r, "F\pi_2"] & FT
\end{tikzcd}$$

must commute.

The above definition of a bisimulation requires that $S \times T$ exist so we will assume going forward that our base categories have products. A bisimulation equivalence is a bisimulation that is also an equivalence relation, a notion that only really makes sense when **Sets** is the base category. Theorems about bisimulations often work out to be very useful for studying coalgebras. I present one here.
Let $G(f)$ be the graph of the set function $f : A \to B$. $G(f) := \{(a, b) \in A \times B \mid (a, b) = (x, f(x))\}$. We can expand this definition to an arrow in a category $\mathcal{C}$ with pullouts.

Definition 5.4. *Given an arrow $f : C \to C'$, $G(f)$ is the pullout of $id_B : B \to B$ and $f : C \to C'$.*

$$\begin{tikzcd}
G(f) \arrow[r, "\pi_1"] \arrow[d, "\pi_2"] & C' \arrow[d, "id_{C'}"] \\
C \arrow[r, "f"] & C'
\end{tikzcd}$$

Recalling that a pullout is the equaliser of arrows $f \circ \pi_1$ and π_2 we can re-frame $G(f)$ to be the object with an arrow $i : G(f) \to C \times C'$ such that:

$$\begin{tikzcd}
& G(f) \arrow[d, "i"] & \\
C \arrow[rr, bend right, "f"'] & C \times C' \arrow[l, "\pi_2"'] \arrow[r, "\pi_f"] & C'
\end{tikzcd}$$

commutes. Of course maintaining its UMP. This perspective highlights the connection of $G(f)$ to the graph of a function over a set.

Lemma 5.5. $C \cong G(f)$

Proof. We just show that C is also a pullout of $G(f)$

$$
\begin{tikzcd}
E \arrow[drr, "e_1"] \arrow[ddr, "e_2"'] & & \\
& C \arrow[r, "f"] \arrow[d, "id_C"] & C' \arrow[d, "id_{C'}"] \\
& C \arrow[r, "f"] & C'
\end{tikzcd}
$$

It holds that $id_{C'} \circ f = f \circ id_C$.
Given any E and arrows $e_1 : E \to C'$ and $e_2 : E \to C$ such that $f \circ e_2 = e_1$. We immediately have that $e_2 = id_C \circ e_2$ and $e_1 = f \circ e_2$. e_2 is unique in this property as it is the only arrow such that $id_C \circ e_2 = e_2$. Thus C is also a pullout of f and $id_{C'}$. So $C \cong G(f)$. Note that f is the arrow such that π_2 is the arrow that fulfils the UMP of C as a pullback since $\pi_1 = f \circ \pi_2$ and $\pi_2 = id_C \circ \pi_2$.

$$
\begin{tikzcd}
G(f) \arrow[drr, "\pi_1"] \arrow[dr, "\pi_2"] \arrow[ddr, "\pi_2"'] & & \\
& C \arrow[r, "f"] \arrow[d, "id_C"] & C' \arrow[d, "id_{C'}"] \\
& C \arrow[r, "f"] & C'
\end{tikzcd}
$$

Then π_2 is the isomorphism through which $C \cong G(f)$.

Theorem 5.6. *[9] (Generalised from working with sets only) Let (S, α_S) and (T, α_T) be two coalgebras. An arrow $f : S \to T$ in $\mathcal{C}$ is a coalgebra homomorphism if and only if $G(f)$ is a bisimulation.*

Proof. Assume that $(G(f), \alpha_{G(f)})$ is a bisimulation between (S, α_S) and (T, α_T) then the below commutes:

$$
\begin{tikzcd}
S \arrow[d, "\alpha_S"] & G(f) \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha_{G(f)}"] & T \arrow[d, "\alpha_T"] \\
FS & F(G(f)) \arrow[l, "F\pi_1"'] \arrow[r, "F\pi_2"] & FT
\end{tikzcd}
$$

Note by Lemma 5.5

$$
\begin{aligned}
\alpha_S \circ \pi_1 &= F\pi_1 \circ \alpha_{G(f)} \\
\alpha_S &= F\pi_1 \circ \alpha_{G(f)} \circ \pi_1^{-1} \\
F\pi_1^{-1} \circ \alpha_S &= \alpha_{G(f)} \circ \pi_1^{-1}
\end{aligned}
$$

And thus:

$$
\begin{aligned}
\alpha_T \circ f &= \alpha_T \circ \pi_2 \circ \pi_1^{-1} \\
&= F\pi_2 \circ \alpha_{G(f)} \circ \pi_1^{-1} \\
&= F\pi_2 \circ F\pi_1^{-1} \circ \alpha_S \\
&= Ff \circ \alpha_S
\end{aligned}
$$

So f is a coalgebra homomorphism.
For the other direction of the implication, assume that f is a coalgebra homomorphism. Then $\alpha_{G(f)} :=$

$F\pi_1^{-1}\alpha_s\pi_1$ is a transition structure that defines $(G(f), \alpha_{G(f)})$ as a coalgebra. It is trivial that $\pi_1 : (G(f), \alpha_{G(f)}) \to (S, \alpha_S)$ is a homomorphism. We see also that:

$$\begin{aligned} F\pi_2 \circ \alpha_{G(f)} &= F\pi_2 \circ F\pi_1^{-1} \circ \alpha_S \circ \pi_1 \\ &= Ff \circ \alpha_S \circ \pi_1 \\ &= \alpha_T \circ f \circ \pi_1 \\ &= \alpha_T \circ \pi_2 \circ \pi_1^{-1} \circ \pi_1 \\ &= \alpha_t \circ \pi_2 \end{aligned}$$

And hence π_2 is also a coalgebra homomorphism.

Considering Lemma 5.5 this theorem feels almost reductive. By $C \cong G(f)$ we are saying f is a homomorphism if it is a homomorphism. However, this theorem is still useful. It highlights to us the reason bisimulation proofs are so helpful. They are a means of re-framing statements. Over **Sets**, this theorem allows us to make statements about a function using its graph which is instead a relation. See [9] which dedicates several pages to results about bisimulations.

6 Limits in Categories of Coalgebras

It is not hard to see that when attempting to prove results for coalgebras it would be ideal to know if $\mathcal{C}_F$ has limits and colimits. In particular the reader may have picked up on the importance of a terminal coalgebra. It is of great value to know when $\mathcal{C}_F$ has a terminal object. Particularly when representing data-types with coalgebras most coalgebra categories without a terminal coalgebras are of little interest. We begin by dealing with colimits.

Theorem 6.1. *([9] details expanded on and generalised from dealing just with sets) $\mathcal{C}_F$ is co-complete if $\mathcal{C}$ is.*

Proof. (Proven in [9] but only in $\mathbf{Sets}_F$) First we show that $\mathcal{C}_F$ has coequalisers.
given homomorphisms $a, b : (S, \alpha_S) \to (T, \alpha_T)$ take $f : T \to W$ as the coequaliser in $\mathcal{C}$. Then

$$\begin{aligned} & f \circ a = f \circ b \\ \implies & Ff \circ Fa = Ff \circ Fb \\ \implies & Ff \circ Fa \circ \alpha_S = Ff \circ Fb \circ \alpha_S \\ \implies & Ff \circ \alpha_T \circ a = Ff \circ \alpha_T \circ b \end{aligned}$$

Appealing the UMP of coequalisers we have an arrow α_W in $\mathcal{C}$ such that $\alpha_W \circ f = Ff \circ \alpha_T$. Then we can define the coalgebra (W, α_W) and it holds that $f : (T, \alpha_T) \to (W, \alpha_W)$ is a homomorphism.

$$\begin{tikzcd} S \arrow[r, shift left, "b"] \arrow[r, shift right, "a"'] & T \arrow[r, "f"] \arrow[dr, "Ff\circ\alpha_T"'] & W \arrow[d, dotted, "\alpha_W"] \\ & & FW \end{tikzcd}$$

Now suppose that there is some homomorphism $e : (T, \alpha_T) \to (E, \alpha_E)$, such that $e \circ a = e \circ b$. We appeal to the coequaliser UMP of $\mathcal{C}$ again to get a unique arrow u in $\mathcal{C}$ such that $u \circ f = e$. f is epic in $\mathcal{C}$ so $a' \circ f = b' \circ f \implies a' = b'$. Since all homomorphisms are arrows of $\mathcal{C}$, the homomorphism f is also epic. So $e = u \circ f$, e and f are homomorphisms and f is epic, then u is a homomorphism by Lemma

5.1. Hence f is also a coequaliser in $\mathcal{C}_F$.
$\mathcal{C}_F$ has coproducts.
Given (S, α_S) and (T, α_T) consider $S + T \in \mathcal{C}$, with injections i_S and i_T. There are arrows $F(i_S) \circ \alpha_S : S \to F(S+T)$ and $F(i_T) \circ \alpha_T : T \to F(S+T)$. Then appealing to the coproduct UMP there is a unique $\alpha_{S+T} : S + T \to F(S+T)$ such that $\alpha_{S+T} \circ i_S = F(i_S) \circ \alpha_S$ and $\alpha_{S+T} \circ i_T = F(i_T) \circ \alpha_T$. Then our injections are homomorphisms onto $(S+T, \alpha_{S+T})$.

$$\begin{tikzcd}
S \arrow[r, "i_S"] \arrow[d, "\alpha_S"] & S+T \arrow[d, dashed, "\alpha_{S+T}"] & T \arrow[l, "i_T"'] \arrow[d, "\alpha_T"] \\
FS \arrow[r, "Fi_S"] & F(S+T) & FT \arrow[l, "Fi_T"']
\end{tikzcd}$$

So we have $(S + T, \alpha_{S+T})$ as our candidate for a coproduct. Suppose there is some (E, α_E) and a pair of homomorphisms $e_1 : (S, \alpha_S) \to (E, \alpha_E)$ and $e_2 : (T, \alpha_T) \to (E, \alpha_E)$. We can appeal to the UMP of a coproduct in $\mathcal{C}$ to get a unique arrow $u : (S + T, \alpha_{S+T}) \to (E, \alpha_E)$ such that $e_j = u \circ i_j$ (I will use j to represent S or T to avoid repeating myself). Now branch significantly of from [9]. We observe that $Fe_j = Fu \circ Fi_j$ and we have the following commutative diagram:

$$\begin{tikzcd}
 & & S+T \arrow[d, "\alpha_{S+T}"] & & \\
S \arrow[urr, "i_S"] \arrow[r, "\alpha_S"] \arrow[ddrr, "e_S"'] & FS \arrow[r, "Fi_S"] \arrow[dr, "Fe_S"'] & F(S+T) \arrow[d, "Fu"] & FT \arrow[l, "Fi_T"'] \arrow[dl, "Fe_T"] & T \arrow[ull, "i_T"'] \arrow[l, "\alpha_T"'] \arrow[ddll, "e_T"] \\
 & & FE & & \\
 & & E \arrow[u, "\alpha_E"'] & &
\end{tikzcd}$$

We observe that given the pair of arrows in the form $Fe_j \circ \alpha_j$.
it holds that

$$\begin{aligned} Fu \circ \alpha_{S+T} \circ i_j &= Fu \circ Fi_j \circ \alpha_j \\ &= Fe_j \circ \alpha_j \end{aligned}$$

and also that

$$\begin{aligned} \alpha_E \circ u \circ i_j &= \alpha_E \circ e_j \\ &= Fe_j \circ \alpha_j \end{aligned}$$

Appealing to the UMP of a coproduct in $\mathcal{C}$ an arrow x such that $Fe_j \circ \alpha_j = x \circ i_j$ is unique. Then $Fu \circ \alpha_{S+T} = x = \alpha_E \circ u$. This demonstrates that u is a homomorphism. Hence $(S + T, \alpha_{S+T})$ is the coproduct of (S, α_S) and (T, α_T).
Therefore $\mathcal{C}_F$ has coequalisers and coproducts and thus it is cocomplete whenever $\mathcal{C}$ is.

This a nice result because it doesn't just inform us about when $\mathcal{C}_F$ is cocomplete, but also that the colimits can be constructed exactly as they are in $\mathcal{C}$.

In the case of limits many choices of F make $\mathbf{Sets}_F$ complete[2]. Here we just summarize [8] since the proof of Watanabe and Power's powerful results are not particularly constructive to building an

[2] [9] actually incorrectly cites [8] on this matter, stating that $\mathbf{Sets}_F$ is always complete

intuition of coalgebras in the way that this report approaches it. The definitions are here to give a limited understanding of when $\mathbf{Sets}_F$ has limits [8].

Definition 6.2. *A functor is accessible if for any infinite ordinal k it acts between k-accessible categories and preserves k-filtered colimits. k-filtered colimits are diagrams with a small category I as their domain. Such that I has a cocone over a diagram $D : J \to I$ whenever the cardinality of J is less than k.*

Theorem 6.3. *[8] if $\mathcal{C}$ is locally presentable and F accessible, then $\mathcal{C}_F$ is complete.*

Sets is locally presentable and many endofunctors on **Sets** are accessible. So it is often true that $\mathbf{Sets}_F$ is complete. This is very helpful in terms of theory crafting. However, the above theorem tells as nothing about how these limits may be constructed, which may hinder our ability to study $\mathbf{Sets}_F$ as a model for a system or data-type.

7 Some More Properties of Coalgebras

Here we introduce some properties a coalgebra may have which will help us when proving a major result about the coinduction principle (section 10).

Definition 7.1. *(S, α_S) is simple if it has no proper quotients. A quotient of (S, α_S) in the language of coalgebra is a coalgebra (T, α_T) and epi homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$. A proper quotient is quotient where f is not an isomorphism.*

Definition 7.2. *The kernel of a homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$ is a subobjet $K \subset S \times S$ with projections such that $f \circ \pi_1 = f \circ \pi_2$ with the UMP that if $Q \subset S \times S$ and has projections such that $f \circ q_1 = f \circ q_2$, Q is a subobject of K.*

In sets this is just $\{(s, s') \mid f(s) = f(s')\}$ [9]. Let π_1 and π_2 be projections from a bisimulation equivalence R. We write $\epsilon_R : S \to S/R$ as the coequaliser of the projections. Then $\epsilon_R(s_1) = \epsilon_R(s_2) \iff (s_1, s_2) \in R$ so R is the kernel of ϵ_R.

Definition 7.3. *A subcoalgebra (V, α_V) of S consists of a subobject V of S with an inclusion arrow i, and a transition structure α_V that makes i a homomorphism. Of note is that i is monic makes Fi monic and the requirement that $Fi \circ \alpha_V = \alpha_S \circ i$ means that α_V is unique.*

A subcoalgebra of interest (that is exclusive to sets) is $\langle s \rangle \subset S$ the set of states of s or the orbit of s. Recall the notation used in section 4; $s \to s'$ says that s transitions to s' ignoring any process on the side like outputs or non-determinism $\langle s \rangle = \{s' \in S \mid s \to s_1 \to \cdots \to s'\}$.

Remark. *The transition structure of $\langle s \rangle$ is just α_S.*

Proof. Considering i as a restriction on id_S $(\alpha_S \circ i)(s) = (Fi \circ \alpha_S)(s)$. We must also check that $s' \in \langle s \rangle \implies \alpha_S(s') \in F\langle s \rangle$. By the definition of $\langle s \rangle$, $s' \to s'' \implies s'' \in \langle s \rangle$ then of course $\alpha_S(s') \in F\langle s \rangle$.

We now look at terminal coalgebras.

Definition 7.4. *A terminal/final coalgebra (P, α_P) is a terminal object in $\mathcal{C}_F$*

As in ordinary category theory terminal coalgebras are unique up to isomorphism. In terms of applications especially for modelling and defining data-types we often only care about final coalgebras. This is highlighted by [3].

Theorem 7.5. *[9] if (P, α_p) is a final coalgebra then α_p is an isomorphism.*

Proof. By the terminality of (P, α_P) there is a homomorphism $f : (FP, F\alpha_p) \to (P, \alpha_p)$.
$\alpha_P : (P, \alpha_P) \to (FP, \alpha_P)$ is also a homorphism since $F(\alpha_p) \circ \alpha_p = F(\alpha_p) \circ \alpha_p$.

$$\begin{tikzcd} P \arrow[r, "\alpha_p"] \arrow[d, "\alpha_P"] & FP \arrow[r, "f"] \arrow[d, "F\alpha_P"] & P \arrow[d, "\alpha_P"] \\ FP \arrow[r, "F\alpha_P"] & FFP \arrow[r, "Ff"] & FP \end{tikzcd}$$

So $f \circ \alpha_P : (P, \alpha_P) \to (P, \alpha_P)$ is a homomorphism (Lemma 5.1). but P is final and we also have $id_{(P,\alpha_P)} : (P, \alpha_P) \to (P, \alpha_P)$ so $id_P = f \circ \alpha_P$.
By f being a homomorphism $\alpha_P \circ f = Ff \circ F\alpha_P = F(f \circ \alpha_P) = id_{FP}$. So indeed $f = \alpha_p^{-1}$.

Corollary 7.5.1. $P \cong FP$ *(immediate)*

Terminal coalgebras are not unique in this property, but by their terminality they are the greatest objects of $\mathcal{C}$ for which this holds. this is encapsulated by the manta: coalgebras (by which we mean final coalgebras) are the greatest fixed points of F. By duality an (initial) algebra induced by F is the least fixed point of F.

This notion is important for defining a data-type where our transition structure acts as a constructor of the data-type. By the transition structure being an isomorphism it may well be viewed as a function $P \to P$ and hence is suitable as a means of recursively defining elements of a data-type. Take $\mathbf{Sets}_F$ where $F(S) = A \times S$ then the final coalgebra is (A^ω, ht). A^ω denotes the set of streams of A and ht is the head tails function. Given a stream $ht(s) = (a, s')$ where a is the first element of s and s' the tail. We note that ht^{-1} adjoins an element to the beginning of a pre-existing stream so it is indeed the constructor of the data-type streams. We can concretely define ht [13] by viewing a stream as a function $f : \mathbb{N} \to A$. Then $ht = \lambda f. < f0, \lambda n.f(n+1) >$ (head gives $f(0)$ tail gives $f(n+1)$).

The reader may note that just having the constructor of streams is of little practical use since it requires that we already have a concrete construction of a stream to begin with. This is addressed by the terminality of (P, α_P) and an application of the coinduction definition principle. Let us construct $(ab)^\omega$ (the stream beginning with a and alternating between a and b). Let $\mathbf{2} = \{0, 1\}$ and consider $(\mathbf{2}, \alpha_\mathbf{2})$ where $\alpha_\mathbf{2}(0) = (a, 1)$ and $\alpha_\mathbf{2}(1) = (b, 0)$. By terminality there is a homomorphism $alt : (\mathbf{2}, \alpha_\mathbf{2}) \to (A^\omega, ht)$. Consider $alt(0)$. It holds that $ht \circ alt(0) = Falt \circ \alpha_\mathbf{2}(0)$:

$$\begin{aligned} Falt \circ \alpha_\mathbf{2}(0) &= Falt(a, 1) \\ &= (a, alt(1)) \\ Falt \circ \alpha_\mathbf{2}(1) &= Falt(b, 0) \\ &= (b, alt(0)) \end{aligned}$$

Then the first element of $alt(0)$ is a, and the second must be b followed by a and so on. Then $alt(0)$ is $(ab)^\omega$. In this sense A^ω is a canonical representation of the way an element of coalgebra transitions. In general a terminal coalgebra along with its homomorphisms is a canonical representation of the dynamics of all coalgebras in the coalgebra category [9]. Our example also encapsulates the notion that streams are a history of signals, as $alt(0)$ is just the emissions of 0 as it is acted on by the transition structure.

We may also use the coinduction definition principle to define functions on streams. As in the last example all that is needed is another coalgebra. Take zipping two streams [9]. Consider the coalgebra $(A^\omega \times A^\omega, z)$ where z is defined as:

$$z(s_1, s_2) = (h(s_1), s_2, t(s_1))$$

Then by A^ω being terminal we have some homomorphism $zip : A^\omega \times A^\omega \to A^\omega$:

$$\begin{tikzcd}
A^\omega & A \times A^\omega \arrow[l, "ht^{-1}"'] \\
A^\omega \times A^\omega \arrow[u, dotted, "zip"] \arrow[r, "z"] & A \times A^\omega \times A^\omega \arrow[u, dotted, "Fzip"']
\end{tikzcd}$$

By considering that zip $= ht^{-1} \circ Fzip \circ z$ we see that

$$\begin{aligned}
zip(s_1, s_2) &= ht^{-1} \circ Fzip(h(s_1), s_2, t(s_1)) \\
&= ht^{-1} \circ (h(s_1), zip(s_2, t(s_1)))
\end{aligned}$$

Recalling that ht^{-1} is the constructor of streams, this is exactly the way that we expect to see zipping streams defined in a language like Haskell. However without the notion of a coalgebra it is not immediate that this is computable, yet for us by defining this coinductively, it is trivial. Such a function must exist by the terminality of A^ω.

In general the coinduction definition principle is the technique of defining homomorphisms just by supplying a domain.

Definition 7.6. *A coalgebra (S, α_S) has the coinduction definition principle if for all other coalgebras (T, α_T) there is at least one homomorphism $f : (T, \alpha_T) \to (S, \alpha_S)$*

A coalgebra with the coinduction principle is sometimes called weakly complete [4]. This is why terminal coalgebras are of such interest, they have the coinduction definition principle. However, a terminal coalgebra has even stronger properties; it has at most one homomorphism onto it from every other coalgebra. We will see in chapter 10 that this also gives terminal coalgebras the coinduction proof principle.

Recall from section 7 that $\mathbf{Sets}_F$ often, but not always has a terminal coalgebra. However, if one is willing to work with classes, $\mathbf{SETS}_F$ the coalgebra also induced by F but over the category of classes does always have a terminal coalgebra. This relates to the notion of accessibility from Theorem 6.3 [8, 7].

For example $\mathbf{Sets}_\mathcal{P}$ where $\mathcal{P}$ is the powerset functor does not terminal object. If it did there would be some set A with $A \cong P(A)$, but this is false for any well-founded set. Aczel's work [4] (if we are willing to replace the axiom of foundation) gives a means of identifying a terminal coalgebra for $\mathbf{Sets}_F$ over a broad class of functors. In the world of sets without the foundation axiom there is indeed a $\mathcal{P}_J$ that is terminal in $\mathbf{Sets}_\mathcal{P}$. If we require our sets to have the foundation axiom we are more restricted in our choices of functors if we would like $\mathbf{Sets}_F$ to have a terminal coalgebra.

8 Special Functors

Let us look at some notable families of functors with the motivation of finding cases in which coalgebra categories may have terminal objects or other interesting properties. We can think of the properties of a coalgebra category as a result of our functor and our base category. Because we are mainly interested in using **Sets** as a base category we will build a catalogue of functors that give $\mathbf{Sets}_F$ some ideal properties.

Definition 8.1. *A weak pullback is like a pull back:*

$$\begin{tikzcd}
E \arrow[dr, dotted, "u"] \arrow[drr, bend left] \arrow[ddr, bend right] & & \\
& P \arrow[r] \arrow[d] & B \arrow[d] \\
& A \arrow[r] & C
\end{tikzcd}$$

with a similar UMP but u need not be unique[3].

Theorem 8.2. *[9] If F preserves weak pullbacks then the kernel of homomorphism is a bisimulation equivalence.*

Proof. First we show that the kernel K of f is the following pullback:

$$\begin{tikzcd}
K \arrow[r, "\pi_1"] \arrow[d, "\pi_2"] & S \arrow[d, "f"] \\
S \arrow[r, "f"] & T
\end{tikzcd}$$

And indeed then $f \circ \pi_1 = f \circ \pi_2$. Suppose there is a subobject of $S \times S$, Q with projections such that $f \circ q_1 = f \circ q_2$. By appealing to the UMP of a pullback $\exists! u : Q \to K$, but we need to confirm that u is mono to say that Q is a subobject of K. Re-framing our pullback as an equaliser of $f \circ p_1$ and $f \circ p_2$ (where p_1 and p_2 are the projections from $S \times S$):

$$\begin{tikzcd}
K \arrow[r, hook, "i"] & S \times S \arrow[r, shift left, "f \circ p_2"] \arrow[r, shift right, "f \circ p_1"'] & T \\
Q \arrow[u, "u"] \arrow[ur, hook, "j"'] & &
\end{tikzcd}$$

i and j are both monic so we observe that

$$\begin{aligned}
& u \circ a = u \circ b \\
\implies & i \circ u \circ a = i \circ u \circ b \\
\implies & j \circ a = j \circ b \\
\implies & a = b
\end{aligned}$$

so u is mono and Q is a subobject of K.
Now we show that the pullback of two homomorphisms $f, g : (S\alpha_S) \to (T, \alpha_T)$ is a bisimulation if F preserves weak pullbacks. If R is a pullback of f and g then it is also a weak pullback of f and g. Hence by assumption FR is a weak pullback of Ff and Fg.

[3] Rutten [9] is particularly interested in functors that preserve weak pull-backs and indeed they have some nice properties.

$$\begin{tikzcd}
R \arrow[dr, dotted, "\alpha_R"] \arrow[drr, bend left, "\alpha_S \circ \pi_1"] \arrow[ddr, bend right, "\alpha_S \circ \pi_2"'] & & \\
& FR \arrow[r, "F\pi_1"] \arrow[d, "F\pi_2"] & FS \arrow[d, "Ff"] \\
& FS \arrow[r, "Fg"] & FT
\end{tikzcd}$$

The outer ring commutes since f and g are homomorphisms:

$$\begin{aligned}
&\alpha_T \circ f \circ \pi_1 = \alpha_T \circ g \circ \pi_2 \\
\implies &Ff \circ \alpha_S \circ \pi_1 = Fg \circ \alpha_T \circ \pi_2
\end{aligned}$$

Then we appeal to the definition of weak pullback to get a map $\alpha_R : R \to FR$ such that $F\pi_1 \circ \alpha_R = \alpha_S \circ \pi_1$ and $F\pi_2 \circ \alpha_R = \alpha_S \circ \pi_2$. Hence the coalgebra (R, α_R) is bisimulation on (S, α_S). By presenting K as a pullback of f with itself we can apply this argument to find a transition structure α_K such that (K, α_K) is a bisimulation of S.
Finally we show that K is an equivalence relation. Going forward this proof must assume that our base category is **Sets**. First $(s, s) \in K$ by $f(s) = f(s)$. Furthermore, $(s, s') \in K \implies f(s) = f(s') \implies (s', s) \in K$. Finally $(s, s') \in K$ and $(s', s'') \in K$ implies that $f(s) = f(s') = f(s'')$ and hence $(s, s'') \in K$.

This ends up being very useful, especially for dualizing the algebraic isomorphism theorems into the theory coalgebras.

Remark. *Every functor we have looked at so far preserves weak pull-backs including $\mathcal{P}$*

Proof. We show $\mathcal{P}$ preserve weak pullbacks. We begin by showing that in **Sets** if P is a pullback then L is a weak pullback if there is a surjection from L to P (assuming axiom of choice).

$$\begin{tikzcd}
L \arrow[dr, "u"] \arrow[drr, bend left, "l_1"] \arrow[ddr, bend right, "l_2"'] & & \\
& P \arrow[r, "\pi_1"] \arrow[d, "\pi_2"] & A \arrow[d, "f"] \\
& B \arrow[r, "g"] & C
\end{tikzcd}$$

Suppose there is some set E and maps $e_1 : E \to A$ and $e_2 : E \to B$. Appealing to the pullback UMP there is a unique e such that $\pi_1 \circ e = e_1$ and $\pi_2 \circ e = e_2$. By the axiom of choice there a map $\bar{e} : E \to L$ that factors e into $e = u \circ \bar{e}$. We must confirm that $l_1 \bar{e} = e_1$ and $l_2 \bar{e} = e_2$

$$\begin{aligned}
e_1 &= \pi_1 \circ e \\
&= \pi_1 \circ u \circ \bar{e} \\
&= l_1 \circ \bar{2}
\end{aligned}$$

Likewise by the same argument $l_2 \bar{e} = e_2$. This converse of the above holds; that if L is a weak pullback and P is a pullback then there is a surjection from L to P.
Now we just show that with $\mathcal{P}$ the image of a weak pullback is also a weak pullback. Again let L be a weak pullback of f and g. Let P be the pullback of f and g. We will also take P' as the pullback of $\mathcal{P}f$ and $\mathcal{P}g$.

$$\begin{tikzcd}
\mathcal{P}L \arrow[dr, two heads, "u"] \arrow[drr, bend left, "l_1"] \arrow[ddr, bend right, "l_2"'] & & \\
& \mathcal{P}P \arrow[r, "\mathcal{P}\pi_1"] \arrow[d, "\mathcal{P}\pi_2"] & \mathcal{P}A \arrow[d, "\mathcal{P}f"] \\
& \mathcal{P}B \arrow[r, "\mathcal{P}g"] & \mathcal{P}C
\end{tikzcd}$$

Given that L surjects onto P with u it follows that $\mathcal{P}u$ is a surjection from $\mathcal{P}L$ to $\mathcal{P}P$. Consider $V_P \in \mathcal{P}P$. Because u is surjective, for any $p \in V_P$ there is an $l \in L$ such that $u(l) = p$. Then $\mathcal{P}u(\{l \in L \mid u(l) \in V_P\}) = V_P$.
Observe that $\mathcal{P}P$ is isomorphic to the set with elements V_P. Where for any $(a, b) \in V_P$, $f(a) = g(b)$. P' is isomorphic to a set with elements $V_{P'}$. Where $V_{P'}$ is the product of $A' \subset A$ and $B' \subset B$. Such that $\mathcal{P}f(A') = \mathcal{P}g(B')$. Then for every $a \in A'$ there is a $b \in B'$ such that $f(a) = b$. Let B^* be the set of all such b. Then $B^* \subset B'$ and $A' \times B^* \subset A' \times B'$. It is also apparent that $A' \times B^*$ is isomorphic to an element of $\mathcal{P}P$. We can also construct $A^* \times B'$ with the same properties but respect to B. Then $A' \times B^* \cup A^* \times B'$ surjects onto $A' \times B'$. So for every element of P' there is a surjection from a union of elements of $\mathcal{P}P$ to that element. Then it follows that there is a surjective map from $\mathcal{P}P$ to P'. Then there is a surjective map from $\mathcal{P}L$ to P' and hence L is also weak pullback of $\mathcal{P}f$ and $\mathcal{P}g$.

Let us consider a subfamily of functors that preserve weak pullbacks called polynomial functors. These are any functors constructed out of products, coproducts, exponentials and fixed set functors (functors that send any set S to a fixed set X), like $FS = 1 + S^B \times A$. All of these functors induce coalgebra categories [9] with terminal objects.
There are many theorems that given a class of functors provide a proof as to why **Sets** coalgebras induced by these functors are complete and have terminal objects [9, 8, 7]. A central theme for most of these relates to the forgetful functor $U : \mathbf{Sets}_F \to \mathbf{Sets}$. The forgetful functor takes a coalgebra (S, α_S) to S and takes a homomorphism to its underlying function.

Theorem 8.3. *If the forgetful functor U has a right adjoint then $\boldsymbol{Sets}_F$ is complete.*

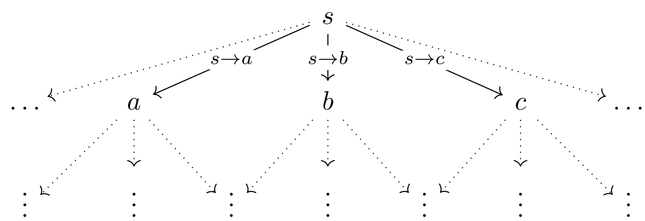
Proof. If U has a right adjoint $R : \mathbf{Sets} \to \mathbf{Sets}_F$, R will preserves limits (RAPL) [12]. By the fact that **Sets** is complete then $\mathbf{Sets}_F$ will be also.

Most theorems proving the existence of a terminal object in $\mathbf{Sets}_F$ are proven by demonstrating that U has a right adjoint. Theorem 6.3 is an application of (the dual of) the special adjoint functor theorem[4] [7].
We now look at bounded functors.

Definition 8.4. *F is bounded if for every coalgebra (S, α_S) in $\boldsymbol{Sets}_F$ and all $s \in (S, \alpha_S)$, the cardinality of $\langle s \rangle$ is bounded.*

It may seem that for any coalgebra (S, α_S), S and hence $\langle s \rangle$ may have as large of a cardinality as we desire. However, given $F = id_{\mathbf{Sets}}$, $\langle s \rangle$ is at most countable since the transitions of s must be a sequence regardless of the size of S. If we think of $\langle s \rangle$ as a tree with root s where branches represent single transitions and nodes are the new states following a transition, we only need the set of branches from every node to be bounded in order for $\langle s \rangle$ to be.

[4]see [12] for a reminder



Rutten proves [9][5] that if F is bounded and preserves weak pullbacks $\mathbf{Sets}_F$ will have a terminal object. I give a different version of this proof as an application of special adjoint functor theorem (dual)[6].

Theorem 8.5. *If F is bounded and preserves weak pullbacks then $\mathbf{Sets}_F$ will have a terminal object*

Proof. We need to show that $\mathbf{Sets}_F$ is cocomplete (shown in Theorem 6.1), locally small, cowell-powered and has a generating set. We also need to show that U (the forgetful functor) preserves colimits.
We show that $\mathbf{Sets}_F$ is locally small. $\mathbf{Sets}$ is locally small and all homomorphisms are arrows in $\mathbf{Sets}$, so $\mathbf{Sets}_F$ is also locally small.
I now show that $\mathbf{Sets}_F$ cowell-powered, that is, is the collection of quotient classes (quotients up to isomorphism) of any coalgebra is a set. I will show that every quotient of (S, α_S) up to isomorphism has just one kernel and this kernel is unique for every quotient. We must appeal to F preserving weak pullbacks so that the kernel is a bisimulation equivalence. First we show that if two quotients have the same kernel they are in the same quotient class. This is because if two epic homomorphisms f, g have the same kernel then $f(s) = f(s') \iff g(s) = g(s')$. We define the map $\phi : \mathrm{cod}(f) \to \mathrm{cod}(g)$ $\phi \circ f(s) = g(s)$. This is well defined because f and g are epic and they have the same kernel. Likewise $\phi^{-1} \circ g(s) = f(s)$. We observe that $\phi \circ f = g$ and $\phi^{-1} \circ g = f$. It follows from Lemma 5.1 that ϕ and ϕ^{-1} are homomorphisms since g is epic. So if two quotients have the same kernel they are in the same quotient class.
If the epic homomorphisms f are g are in the same quotient class then there is some isomorphism ϕ such that $\phi \circ f = g$ and $\phi^{-1} \circ f = g$.
Then

$$\begin{aligned} & f(s) = f(s') \\ \iff & \phi \circ f(s) = \phi \circ f(s') \\ \iff & g(s) = g(s') \end{aligned}$$

hence the kernel of f and g is the same.
So there is a single unique kernel for every quotient. For every kernel (K, α_K) every set K is contained in $\mathcal{P}(S \times S)$. α_K can be considered a restriction of map from $S \times S$ to $F(S \times S)$. By $\mathbf{Sets}$ being locally small the maps from $S \times S$ to $F(S \times S)$ can be contained in a set. So both components of a kernel can be contained in a set. Hence there is a set that contains every kernel. Since every quotient class has a single unique kernel associated with it we can then contain every quotient class of (S, α_S) in a set.
Now I show that $\mathbf{Sets}_F$ has a generating set. By the cardinality of every $\langle s \rangle$ being bounded we have a V such that there is an injective map from every $\langle s \rangle$ onto V. I show that

$$\{(U, \gamma) \mid U \subset V,\ \gamma : U \to FU\}$$

is a generating set for $\mathbf{Sets}_F$. Consider two distinct $g, f : (X, \alpha_X) \to (Y, \alpha_Y)$. There is some $x \in X$ such that $f(x) \neq g(x)$. Then consider $i : (\langle x \rangle, \alpha_X) \to (X, \alpha_X)$ the inclusion of $(\langle x \rangle, \alpha_X)$ into (X, α_X).

[5]see page 45 and Theorem 10.3 and 10.4
[6]As I understand this proof is novel but the approach is inspired by [7] Theorem 1.2

Since $(f \circ i)(x) \neq (g \circ i)(x)$ it follows that $f \circ i \neq g \circ i$. Because F is bounded there is an injective map $u : \langle x \rangle \to V$. Let us refer to $u(\langle x \rangle)$ as U. We may factor u as $u = j \circ s$, where j is the inclusion from U to V and s is an isomorphism from $\langle x \rangle$ to U. Define $\gamma := Fs \circ \alpha_X \circ s^{-1}$.

$$\begin{tikzcd} \langle x \rangle \arrow[r, "\cong s"] \arrow[d, "\alpha_X"] & U \arrow[d, dotted, "\gamma"] \\ F\langle x \rangle \arrow[r, "\cong Fs"'] & FU \end{tikzcd}$$

Let us show that s is an isomorphic homomorphism from $(\langle x \rangle, \alpha_X)$ to (U, γ)

$$\begin{aligned} \gamma \circ s &= Fs \circ \alpha_X \circ s^{-1} \circ s \\ &= Fs \circ \alpha_X \\ Fs^{-1} \circ \gamma &= Fs^{-1} \circ Fs \circ \alpha_X \circ s^{-1} \\ &= \alpha_X \circ s^{-1} \end{aligned}$$

and evidently $s \circ s^{-1} = id_U,\ s^{-1} \circ s = id_{\langle x \rangle}$. Then there is a homomorphism $i \circ s^{-1} : (U, \gamma) \to (X, \alpha_X)$. Because s^{-1} is an isomorphism:

$$\begin{aligned} & f \circ i \neq g \circ i \\ \implies & f \circ i \circ s^{-1} \neq g \circ i \circ s^{-1} \end{aligned}$$

We also observe that $(U, \gamma) \in \{(U, \gamma) \mid U \subset V,\ \gamma : U \to FU\}$. So $\{(U, \gamma) \mid U \subset V,\ \gamma : U \to FU\}$ is the generating set $\mathbf{Sets}_F$.
Now we only need to show that the forgetful functor preserves small colimits. In proving Theorem 6.1 we demonstrated that any colimit in $\mathbf{Sets}_F$ is the same colimit in **Sets**. Of course we had to define transition structures for our colimit, but these transition structures are removed by the action of the forgetful functor. So any colimit in $\mathbf{Sets}_F$ is returned to the same colimit in **Sets** by the forgetful functor. Hence the forgetful functor preserves colimits.
Therefore if F is bounded and preserves weak pullbacks then $\mathbf{Sets}_F$ is complete and has a terminal object.

We now briefly discuss comonad coalgebras [12, 15, 10]. Given an adjunction $F \dashv U$ we can compose the two functors as so $F \circ U =: G$ to get an endofunctor ($G : \mathcal{C} \to \mathcal{C}$). We call G a comonad. An equivalent definition is.

Definition 8.6. *An endofunctor G is a comonad if it is equipped with the natural transformations*

$$\begin{array}{ll} \epsilon : G \to 1 & \textit{counit} \\ \delta : G \to G^2 & \textit{coproduct} \end{array}$$

Such that for any $S \in \mathcal{C}$

$$\begin{aligned} \delta_{GS} \circ \delta_S &= G\delta_S \circ \delta_S \\ G\epsilon_S \circ \delta_S &= id_{GS} = \epsilon_{GS} \circ \delta_S \end{aligned}$$

Within $\mathcal{C}_G$ there is subcategory consisting of coalgebras that reflect this structure internally. These are coalgebras (S, α_S) such that

$$\begin{aligned} \alpha_S \circ \delta_S &= G\alpha_S \circ \alpha_S \\ \epsilon_S \circ \alpha_S &= id_S \end{aligned}$$

A comonad coalgebra category is just this subcategory. Its arrows are homomorphisms between comonad coalgebras. The structure that we are imposing on our coalgebras is quite natural. Recall how in Section 2 we induced an algebra category that contained all groups. The groups in this category were precisely the monad algebras. We don't generally identify familiar constructions when we consider comonad coalgebras like we do with monad algebras because coalgebras do not occur naturally in the same ways that algebras do. However, there are still many applications where we only care about comonad coalgebras.

One such application of these properties is in modeling context dependant functions [11]. These are functions used in programming where the computation does not depend just on the input but some wider context. This is done using a different category called a coKleisli category. However, the idea behind a coKleisli category is very similar to a coalgebra category. We see how our comonad coalgebras might apply to cotext dependant functions. Our transition structure under this framing represents contextualisation of the data. The action of ϵ_S makes this forgettable, separating the context of the data from the data itself. Coalgebra homomorphisms now represent context dependant functions as they must respect this context. An example of context based computation that we know how to model with coalgebras is streams. Streams are often thought of as a history of signal emissions. The next emission of a signal may well be dependant on all the past emissions not just the most recent one. Thus the tail of the stream is the context even if we might think of a function as one on just the head [11].

9 Coalgebras in the Language of Algebra

We now compare coalgebras with algebras. I present the first isomorphism theorem of coalgebras just for coalgebras over **Sets**. For the first isomorphism theorem to hold in $\mathbf{Sets}_F$, F must preserve weak pullbacks.

Theorem 9.1. *$f : (S, \alpha_S) \to (T, \alpha_T)$ factorizes in $\boldsymbol{Sets}_F$ as $f = i \circ \bar{f}$:*

$$\begin{tikzcd}
(S, \alpha_S) \arrow[rr, "f" description] \arrow[dr, two heads, "\bar{f}" description] \arrow[ddr, bend right, two heads, "\epsilon_K"'] & & (T, \alpha_T) \\
& f(S) \arrow[ur, hook, "i" description] & \\
& S/K \arrow[u, "\cong"] \arrow[uur, bend right, hook, "\mu"'] &
\end{tikzcd}$$

Hooked tails represent a mono, I have omitted transition structures.

Proof. [9] proves this is full. I will just show that $f(S) \cong S/K$ in $\mathbf{Sets}_F$.
First I show that $(f(S), \alpha_T)$ is a subobject of (T, α_T). Let i be the inclusion, that is $i(t) = t$. Then it is apparent that $(\alpha_T \circ i)(t) = (Fi \circ \alpha_T)(t)$. We must also verify that $\alpha_T(t) \in F(f(S))$. We note that $t = f(s)$ for some $s \in S$.
Then

$$\begin{aligned}
\alpha_T(t) &= (\alpha_T \circ f)(s) \\
&= (Ff \circ \alpha_S)(s) \\
\implies \alpha_T &\in (Ff \circ \alpha_S)(S) \\
\implies \alpha_T &\in F(f(S))
\end{aligned}$$

So $(f(S), \alpha_T)$ is a subcoalgebra of (T, α_T).
We now show that $f(S) \cong S/K$. Because F preserves weak pull backs K is bisimulation equivalence. By definition ϵ_K is the coequaliser of the projections of K (π_1 and π_2). We define the homomorphism $\bar{f}$ as $\bar{f}(s) = f(s)$ for all s. We will show that $\bar{f}$ is also a coequaliser of π_1 and π_2 in **Sets**. Since K is the kernel of f, $f \circ \pi_1 = f \circ \pi_2$ and so $\bar{f} \circ \pi_1 = \bar{f} \circ \pi_2$. Suppose there is some function $e : S \to E$ so that $e \circ \pi_1 = e \circ \pi_2$.
Then

$$\begin{aligned} &\bar{f}(s) = \bar{f}(s') \\ \implies &(s, s') \in K \\ \implies &(e \circ \pi_1)(s, s') = (e \circ \pi_2)(s, s') \\ \implies &e(s) = e(s') \end{aligned}$$

Consider $u : f(S) \to E$ defined as $(u \circ \bar{f})(s) = e(s)$ for $s \in S$, by above this is well-defined. We observe that $u \circ \bar{f} = e$. u is also unique in this property as suppose $u' \circ \bar{f} = e$. Then $(u' \circ \bar{f})(s) = e(s)$ and so $u' = u$. We know by Theorem 6.1 that $\bar{f} : (S, \alpha_S) \to (f(S), \gamma)$ is a coequaliser of π_1 and π_2 in $\mathbf{Sets}_F$. I will show that $\gamma = \alpha_T$ using the same technique as Theorem 6.1. We know that $F\bar{f} \circ \alpha_S \circ \pi_1 = F\bar{f} \circ \alpha_S \circ \pi_2$. Appealing the UMP of a coequaliser there is a unique $x : f(S) \to F(f(S))$ such that $F\bar{f} \circ \alpha_S = x \circ \bar{f}$. However $F\bar{f} \circ \alpha_S = \gamma \circ \bar{f}$ and $F\bar{f} \circ \alpha_S = \alpha_T \circ \bar{f}$. Hence $\gamma = \alpha_T$. Then $\bar{f} : (S, \alpha_S) \to (f(S), \alpha_S)$ is also a coequaliser of π_1 and π_2. Therefore $f(S) \cong S/K$.

Rutten [9] gives us the second and third isomorphism theorems. We will use the second isomorphism theorem later which I have presented below.

Theorem 9.2. *Given a homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$ and a bisimulation equivalence R that is a subset of the kernel of f, f factorizes uniquely as $f = \bar{f}\epsilon_R$.*

These theorems may encourage us to try and consider coalgebras in the way we might think of concrete algebras, as a set of manipulations defined with equational rules. Recalling the functor we used induce the algebra category for groups, consider the coalgebra induced by it. We will think of a transition structure of a coalgebra in this category as consisting of three parts.

$$\begin{tikzcd} & S \arrow[dl, dashed, "u^*"'] \arrow[d, dashed, "(-)^{-1}"] \arrow[dr, dashed, "\circ^*"] & \\ 1 & S & S \times S \end{tikzcd}$$

An anti unit, inverse and composition. We have lost our unique identity, the ability to take the inverse of any element and our binary operation now acts as a forced factorization. This does not make sense if we try to think about it like we would a traditional algebra. Instead this example consolidates Rutten's idea that coalgebras represent systems, not manipulations. Our elements do not interact, instead they are manipulated by our transition structure. Thus they better represent the dynamics of a system on its elements.

10 The Coinduction Proof Principle

Peter Aczel introduced the coinduction proof principle with his work on non-well-founded sets [4] in 1988. The coinduction proof principle is the dual of the well known induction principle which applies to algebras. However the discovery of the coinduction proof principle was not as immediate as considering the dual of the induction principle. This is because coinduction proof principle as it is practically used is logically equivalent to the dual of the induction proof principle, but this equivalence is not immediate [9]. Of note is that Aczel presented this in his lecture notes on his theory of Non-Well-Founded Sets in the field of Set Theory. This enlightens us to a new perspective on coalgebras.

Non-Well-Founded sets are an alternative construction to the sets arising from the ZFC axioms. The foundation axiom is replaced with an anti-foundation axiom. This removes the requirement of traditional set theory that if one were to take an element of a set and then an element of that set and so on, they must eventually reach the empty set. Non-well-founded sets do not have this requirement. The set $\Omega = \{\Omega\}$ is a perfectly valid set in Aczel's construction. Coalgebras often have a similar structure to a non-well-founded set. Take a coalgebra over the category of **Sets**

$$(S, \alpha_S : S \to S)$$
$$s \to s'$$

So s transitions to s'. We can think s' as an element of s. If $s' \to s''$ then s'' is an element of s'. Due to the nature of our transition structure every set contains another set just like a non-well-founded set.

We now begin to unravel the coinduction proof principle.

Definition 10.1. *In* $\boldsymbol{Sets}_F$ *the coalgebra* (S, α_S) *has the coinduction proof principle if every bisimulation of* S *is a subset of* Δ_S*.* Δ_S *is the set* $\{(s, s) \in S \times S \mid s \in S\}$

We may generalise this definition. Considering that $R \subset \Delta_S$ occurs just when the projection arrows are equal then we say that any coalgebra (S, α_S) has the coinduction principle whenever this holds. Appealing to the product UMP if (S, α_S) has the coinduction proof principle, any arrow $e : (E, \alpha_E) \to (S, \alpha_S)$ will uniquely factor through R. Then R must reflect the structure of (S, α_S). This is precisely what the coinduction proof principle is used for. Proving statements about a coalgebra (S, α_S) by analysing a bisimulation on S.
An alternative definition is to instead say that a coalgebra has the coinduction proof principle if it is strongly extensional [4]. Where a weakly complete coalgebra (W, α_W) has at least one homomorphism $f : (S, \alpha_S) \to (W, \alpha_W)$ for every (S, α_S); a strongly extensional coalgebra (H, α_H) has at most one homomorphism $f : (S, \alpha_S) \to (H, \alpha_H)$ for every (S, α_S). Our two generalised definitions for the coinduction proof principle are equivalent. This further highlights the importance of terminal coalgebras as they are the only objects with the coinduction proof and definition principle.

Theorem 10.2. *[9] (with many details expanded on) Assuming that* F *preserves weak pullbacks TFAE and working with set co-algebras:*

$$S \text{ is simple} \tag{1}$$
$$S \text{ satisfies the coinduction proof principle (Definition 10.1)} \tag{2}$$
$$\Delta_S \text{ is the only bisimulation equivalence on } S \tag{3}$$
$$S \text{ is strongly extensional} \tag{4}$$
$$\epsilon : S \to S/\backsim \text{ is an isomorphism where } \backsim \text{ is the greatest bisimulation of } S \tag{5}$$

Proof. **1** $\implies$ **3** Let R be a bisimulation equivalence of S, and let f be the coequaliser of the projections from R, π_1 and π_2. Then f is epic so by assumption it must be an isomorphism. Then f is also monic.

Since f is the coequaliser of π_1 and π_2, $f \circ \pi_1 = f \circ \pi_2$ but since f is monic $\pi_1 = \pi_2$. Then R is a subset of Δ_S. It follows that if $(s_1, s_2) \in R$ then $s_1 = s_2$. Since R is an equivalence class it must contain a relation for every s so $R = \{(s,s) \in S \times S\} = \Delta_S$.

3 $\implies$ **1** Consider an epic homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$. Because F preserves weak pullbacks $K(f)$ is a bisimulation equivalence. Then $K(f) = \Delta_S$ by assumption. The projections from Δ_S, π_1 and π_2 are equal so id_S is the coequaliser of the projections. Hence $S \cong S/\Delta(S)$. Because f is epic in $\mathbf{Sets}_F$ it is a surjective function in **Sets** ([9] proposition 4.7) so $f(S) = T$. If we apply the first isomorphism theorem.

$$\begin{tikzcd} S \arrow[rr, "f"] \arrow[dr, "\cong"] & & f(S) = T \\ & S/\Delta_S \arrow[ur, "\cong"] & \end{tikzcd}$$

we see that f is an isomorphism. Hence (S, α_S) is simple.

2 $\implies$ **4** Given two homomorphism $f, g : (T, \alpha_T) \to (S, \alpha_S)$ we may define Q as $\{(f(t), g(t)) \mid \forall t \in T\} \subset S \times S$. We observe that $Q = G(f)^{-1} \circ G(g)$. Theorem 5.6 tells us that $G(f)$ and $G(g)$ are bisimulations. $G(f)^{-1}$ is $G(f)$ with the first and second element of each tuple has been swapped (i.e. $(s,t) \in G(f) \iff (t,s) \in G(f)^{-1}$). It follows that the projections of $G(f)^{-1}$ are the projections of $G(f)$ but swapped. Hence $G(f)^{-1}$ is a bisimulation. Theorem 5.4 from [9] tells us that the composition of bisimulations is a bisimulation. Then Q is a bisimulation. Thus Q is a subset of Δ_S by assumption. Therefore $f(t) = g(t)$ for every $t \in T$ and hence $f = g$.

4 $\implies$ **2** Let R be a bisimulation of (S, α_S) then the projections of R, π_1 and π_2 are homomorphisms from R to (S, α_S). By assumption $\pi_1 = \pi_2$. Hence $R \subset \Delta_S$ and so (S, α_S) has the coinduction proof principle.

2 $\implies$ **3** Let be R be a bisimulation equivalence on (S, α_S). Then $R \subset \Delta_S$ by assumption. Then $(s, s') \in R$ implies that $s = s'$. For are to be a bisimulation equivalence it must have a relation for every element of S then $R = \{(s,s) \in S \times S\} = \Delta_S$.

3 $\implies$ **2** (contrapositive) Let R be a bisimulation of (S, α_S) such that $R \not\subset \Delta_S$. Let f be the coequaliser of the projections from R. Let K be the kernel of f. Because F preserves weak pullbacks K is a bisimulation equivalence. Since f is the coequaliser of the projections from R, if $(s, s') \in R$ then $f(s) = f(s')$ and so $(s, s') \in K$. Then $R \subset K$ and thus $K \not\subset \Delta_S$. By appealing to the contrapositive it follows that if Δ_S is the only bisimulation equivalence of S then all bisimulations are a subset of Δ_S.

1 $\implies$ **5** $\epsilon : S \to S/\backsim$ is the coequaliser of the projections from $\backsim$. Then ϵ is epic and by assumption ϵ is an isomorphism.

5 $\implies$ **3** Let R be any equivalence relation. Then $R \subset \backsim$. By applying the second isomorphism theorem we can uniquely factorise ϵ and $\epsilon = \bar{\epsilon} \circ \epsilon_R$. Consider this factorization in **Sets**. Because ϵ is an isomorphic homomorphism it is a bijective function. It follows that ϵ_R must be an injection. Thus if $\epsilon_R(s) = \epsilon_R(s')$ it must be that $s = s'$. Therefore if $(s, s') \in R \iff s = s'$ then $R = \Delta_S$.

The roles of this theorem is to make it easy to identify a coalgebra that possesses the coinduction proof principle. I conclude this paper with an example of how we might make use of the coinduction proof principle. We will build on our examples from Section 7 to show that $zip((a)^\omega, (b)^\omega) = (ab)^\omega$ [9]. The stream $(a)^\omega$ is the constant stream of a. Consider $R \subset A^\omega \times A^\omega$. $R = \{(zip((a)^\omega, (b)^\omega), (ab)^\omega), (zip((b)^\omega, (a)^\omega), (ba)^\omega)\}$. Because (A^ω, ht) is terminal in $\mathbf{Sets}_F$ (where $F(S) = A \times S$), (A^ω, ht) is strongly extensional, and thus has the coinduction proof principle. Therefore all we need to show is that R is a bisimulation equivalence. Consider the action of ht on the elements of the tuples in R:

$$\begin{aligned}
ht(zip((a)^\omega, (b)^\omega)) &= (a, zip((b)^\omega, (a)^\omega) \\
\implies zip((a)^\omega, (b)^\omega) &\underset{a}{\rightarrow} zip((b)^\omega, (a)^\omega) \\
ht((ab)^\omega) &\underset{a}{\rightarrow} (ba)^\omega \\
ht(zip((b)^\omega, (a)^\omega)) &= (b, zip((a)^\omega, (b)^\omega) \\
\implies zip((b)^\omega, (a)^\omega) &\underset{b}{\rightarrow} zip((a)^\omega, (b)^\omega) \\
ht((ba)^\omega) &\underset{b}{\rightarrow} (ab)^\omega
\end{aligned}$$

So on the first and second element of each tuple in R ht emits the same signal and takes them to the first and second element of the other tuple. Hence we can define the coalgebra (R, ht). This is a bisimulation as if we consider each projection they are just inclusion homomorphisms of $\langle zip((a)^\omega, (b)^\omega)\rangle$ and $\langle (ab)^\omega \rangle$. Therefore by appealing to the coinduction proof principle, R is a subset of Δ_S and so $(zip((a)^\omega, (b)^\omega) = (ab)^\omega$ and $zip((b)^\omega, (a)^\omega) = (ba)^\omega$.

This concludes my short introduction into the theory of coalgebras and their applications. I would like to thank Ranald Clouston for introducing me to Category Theory and coalgebras. This report specifically benefited from Ranald Clouston introducing me to coalgebras and Rutten's work. Furthermore, I benefited from being assisted in understanding several concepts.

References

[1] Saunders Mac Lane. *Categories for the working mathematician.* 6th ed. SPRINGER-VERLAG, 1971. ISBN: 9780387900353,0-387-90035-7.

[2] G.D. Plotkin M. Hennessy. "Full abstraction for a simple parallel programming language". In: *Lecture Notes in Computer Science* (1979).

[3] Michael A. Arbib and Ernest G. Manes. "Parametrized data types do not need highly constrained parameters". In: *Information and Control* (1982).

[4] Peter Aczel. *Non-Well-Founded Sets.* Center for the Study of Language and Information (CSLI), 1988.

[5] Peter Aczel and Nax Paul Mendler. "A Final Coalgebra Theorem". In: *Category Theory and Computer Science* (1989).

[6] Philip Wadler. *Recursive types for free!* 1990. URL: `https://homepages.inf.ed.ac.uk/wadler/papers/free-rectypes/free-rectypes.txt`.

[7] Michael Barr. "Terminal coalgebras in well-founded set theory". In: *Theoretical Computer Science* 114.2 (1993), pp. 299–315. ISSN: 0304-3975. DOI: `https://doi.org/10.1016/0304-3975(93)90076-6`. URL: `https://www.sciencedirect.com/science/article/pii/0304397593900766`.

[8] John Power and Hiroshi Watanabe. "An axiomatics for categories of coalgebras". In: *Electronic Notes in Theoretical Computer Science* 11 (1998). CMCS '98, First Workshop on Coalgebraic Methods in Computer Science, pp. 158–175. ISSN: 1571-0661. DOI: `https://doi.org/10.1016/S1571-0661(04)00057-X`. URL: `https://www.sciencedirect.com/science/article/pii/S157106610400057X`.

[9] J.J.M.M. Rutten. "Universal coalgebra: a theory of systems". In: *Theoretical Computer Science* 249.1 (2000). Modern Algebra, pp. 3–80. ISSN: 0304-3975. DOI: `https://doi.org/10.1016/S0304-3975(00)00056-6`. URL: `https://www.sciencedirect.com/science/article/pii/S0304397500000566`.

[10] Ranald Clouston. "Covarieties of Coalgebras: Comonads and Coequations". In: *Lecture Notes in Computer Science* (2005).

[11] Tarmo Uustalu and Varmo Vene. "Comonadic Notions of Computation". In: *Electronic Notes in Theoretical Computer Science* 203.5 (2008). Proceedings of the Ninth Workshop on Coalgebraic Methods in Computer Science (CMCS 2008), pp. 263–284. ISSN: 1571-0661. DOI: `https://doi.org/10.1016/j.entcs.2008.05.029`. URL: `https://www.sciencedirect.com/science/article/pii/S1571066108003435`.

[12] S. Awodey. *Category Theory.* Oxford Logic Guides. OUP Oxford, 2010. ISBN: 9780199587360. URL: `https://books.google.com.au/books?id=UCgUDAAAQBAJ`.

[13] Venanzio Capretta. "Coalgebras in functional programming and type theory". In: *Theoretical Computer Science* 412.38 (2011). CMCS Tenth Anniversary Meeting, pp. 5006–5024. ISSN: 0304-3975. DOI: `https://doi.org/10.1016/j.tcs.2011.04.024`. URL: `https://www.sciencedirect.com/science/article/pii/S0304397511003227`.

[14] Bartosz Milewski. *Understanding F-Algebras.* 2013. URL: `https://www.schoolofhaskell.com/user/bartosz/understanding-algebras#f-algebras`.

[15] nLab. *comonad.* Online maths wiki page. Last updated May 2024. 2013. URL: `https://ncatlab.org/nlab/show/comonad`.