Constructive Final Semantics of Finite Bags

Philipp Joram
Department of Software Science, Tallinn University of Technology, Estonia

Niccolò Veltri
Department of Software Science, Tallinn University of Technology, Estonia

Abstract

Finitely-branching and unlabelled dynamical systems are typically modelled as coalgebras for the finite powerset functor. If states are reachable in multiple ways, coalgebras for the finite bag functor provide a more faithful representation. The final coalgebra of this functor is employed as a denotational domain for the evaluation of such systems. Elements of the final coalgebra are non-wellfounded trees with finite unordered branching, representing the evolution of systems starting from a given initial state.

This paper is dedicated to the construction of the final coalgebra of the finite bag functor in homotopy type theory (HoTT). We first compare various equivalent definitions of finite bags employing higher inductive types, both as sets and as groupoids (in the sense of HoTT). We then analyze a few well-known, classical set-theoretic constructions of final coalgebras in our constructive setting. We show that, in the case of set-based definitions of finite bags, some constructions are intrinsically classical, in the sense that they are equivalent to some weak form of excluded middle. Nevertheless, a type satisfying the universal property of the final coalgebra can be constructed in HoTT employing the groupoid-based definition of finite bags. We conclude by discussing generalizations of our constructions to the wider class of analytic functors.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Constructive mathematics

Keywords and phrases finite bags, final coalgebra, homotopy type theory, Cubical Agda

Digital Object Identifier 10.4230/LIPIcs.ITP.2023.20

Supplementary Material *Software (Agda Code)*: `github.com/phijor/agda-cubical-multiset` archived at `swh:1:snp:3c33a341583333a888a148d4a91c08b94e404482`

Funding This work was supported by the Estonian Research Council grant PSG749 and the ESF funded Estonian IT Academy research measure (project 2014-2020.4.05.19-0001).

1 Introduction

Coalgebras are functions of the form $c : S \to FS$, where S is a set of states and F is a functor specifying a certain class of collections of states [23, 14]. For example, FS could be lists over S, bags (*i.e.* multisubsets) or subsets of S (possibly with some cardinality restrictions), wellfounded trees with leaves or nodes in S, or probability distributions over S. The coalgebra c describes the dynamics of a transition system or an automaton: to each state $s : S$, the function c associates the collection of states $c\,s : FS$ that are reachable from s in one step. The choice of collection functor F is dictated by the specific flavor of non-determinism that is specified by the transition relation. Does the order or multiplicity of reachable states matter? Is the choice of a new state probabilistic? Does the transition relation additionally depend on a set of labels, weights or actions?

The denotational semantics of a transition system $c : S \to FS$ is typically given in terms of the *final coalgebra* L_F of the functor F, which consists of non-wellfounded trees with branching specified by F. When F is the list functor, each tree has a finite and ordered collection of subtrees. If F is the finite bag functor, the order of subtrees does not matter,

and if F is the finite powerset functor, multiplicity of subtrees does not matter either. The interpretation of a state $s : S$ in L_F is the possibly-infinite tree obtained by "running" the coalgebra c with s as initial state. As such, it gives a complete description of the evolution of the system c starting from state s.

The theory of dynamical systems as coalgebras [23, 14], and in particular the formal description of final coalgebras [4, 2, 32], with the associated notion of bisimilarity and behavioural equivalence of states, is traditionally developed in a set-theoretic framework with reasoning based on classical logic. In this work, we propose to study the theory of coalgebras in a framework based on constructive logic, more specifically in homotopy type theory (HoTT) [26]. The use of a constructive metatheory is beneficial for the development of *formal denotational semantics* of dynamical systems and programming languages, often centered on the notions of final coalgebra and bisimilarity [27], in proof assistants based on variants of Martin-Löf type theory, such as Agda, Coq, Idris and Lean. The specific choice of HoTT is motivated by its expressiveness and innovative features, higher inductive types (HITs) and the univalence principle, which are crucial ingredients for faithfully representing a variety of collection functors F and reasoning up to equivalent presentations of F.

Specific constructions of final coalgebras for a selection of functors, performed *internally* in HoTT, already exist in the literature. Ahrens *et al.* [3] presented a construction of *M-types*, *i.e.* final coalgebras of polynomial functors. They show that, for a polynomial functor F, the traditional set-theoretic construction of its M-type as the ω-limit of the chain

$$1 \xleftarrow{!} F1 \xleftarrow{F(!)} F^2 1 \xleftarrow{F^2(!)} F^3 1 \xleftarrow{F^3(!)} \cdots \tag{1}$$

(with 1 being the unit type and ! the unique map into 1) can be ported without major complications to the setting of HoTT. Veltri [29] examined various constructions of the final coalgebra of the finite powerset functor, which is known to not be definable as an ω-limit [2]. Worrell proposed a set-theoretic construction as a $(\omega + \omega)$-limit [32], but Veltri showed that this cannot be ported to the constructive setting of HoTT: Worrell's construction defines the final coalgebra of the finite powerset functor if and only if the *lesser limited principle of omniscience* (LLPO) holds, which is a constructive taboo [6].

We extend this line of work by studying the final coalgebra of the *finite bag functor*. This is an intermediate situation between finitary polynomial functors, such as the one delivering lists, and general finitary functors, such as the one delivering finite subsets. It also serves as a representative starting point for a constructive analysis of Joyal's *analytic functors* [15] and their final semantics. In type theory, analytic functors arise from *quotient containers* [1] and encompass many datatypes with symmetries, such as finite bags, unordered pairs and cyclic lists [34, 33].

Following the recent work of Choudhury and Fiore [8], we define and compare various implementations of the type of finite bags in HoTT. Choudhury and Fiore give two equivalent presentations of finite bags as HITs: as free commutative monoids and as lists modulo swapping of adjacent entries. We add an equivalent presentation of finite bags as an analytic functor FMSet: an element of $\mathsf{FMSet}\, X$ is a pair of a natural number n (its size) and an equivalence class of functions typed $\mathsf{Fin}\, n \to X$ picking an element of X for each $k < n$. Two functions $v, w : \mathsf{Fin}\, n \to X$ belong to the same equivalence class if there merely exists an equivalence $\sigma : \mathsf{Fin}\, n \to \mathsf{Fin}\, n$ such that $v = w \circ \sigma$. The type $\mathsf{FMSet}\, X$ is always a set (in the sense of HoTT, *i.e.* a type with at most one identification between any two terms), since it employs set-quotienting. Similarly, the HITs of Choudhury and Fiore are sets. Following [16], finite bags can alternatively be defined as a polynomial functor Bag returning a groupoid (in the sense of HoTT, *i.e.* a type whose equality types are sets) instead of a set. In this case,

an element of $\mathsf{Bag}\,X$ is a pair consisting of a (Bishop-)finite type Y and a function from Y to X. The set-based and the groupoid-based definitions of bags are appropriately related by the set-truncation construction: $\|\mathsf{Bag}\,X\|_2 \simeq \mathsf{FMSet}\,X$.

We investigate 3 constructions of the final coalgebra of the finite bag functor:

1. Working with the set-based functor FMSet, we try to replicate the classical set-theoretic construction as an ω-limit of (1) in our constructive setting. We show that this cannot be directly performed in HoTT without introducing some form of classical logic, an issue already spotted in the case of the finite powerset functor [29]. Formally, we show that FMSet weakly preserves the ω-limit of (1), but strong preservation of this limit implies LLPO.
2. The list functor admits a final coalgebra $\mathsf{L}_{\mathsf{List}}$ in HoTT [3] and classically an appropriate quotient of the latter, by a relation Bisim identifying non-wellfounded trees which differ in the order of their subtrees, delivers the final coalgebra of the finite bag functor. This construction is also inherently classical: attempting to define a FMSet-coalgebra structure on $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$, *i.e.* a function of type $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim} \to \mathsf{FMSet}(\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim})$, by directly lifting the List-coalgebra structure of $\mathsf{L}_{\mathsf{List}}$, implies LLPO. We point out that this issue already appears in the category of *setoids* [5], before effectively forming the set-quotient $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$. We were able to prove that $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$ is the final FMSet-coalgebra only under the assumption of the axiom of choice.
3. The groupoid-based polynomial functor Bag admits a final coalgebra $\mathsf{L}_{\mathsf{Bag}}$ as the ω-limit of (1), a result arising directly from the work of Ahrens *et al.* [3]. $\mathsf{L}_{\mathsf{Bag}}$ is a groupoid, not a set. One might wonder if the set-truncation $\|\mathsf{L}_{\mathsf{Bag}}\|_2$ is a good candidate for the final FMSet-coalgebra. We show that it is a fixpoint of FMSet, but we were able to prove that it is the final coalgebra only under the assumption of two variants of the axiom of choice.

We do not yet know whether the uses of choice in the last two constructions are also necessary. Nevertheless, the set-truncation $\|\mathsf{L}_{\mathsf{Bag}}\|_2$ can be practically employed as denotational domain for transition systems with a *(Bishop-)finite* set of states: given a coalgebra $c : S \to \mathsf{FMSet}\,S$ where S is finite, there exists a unique coalgebra morphism from S to $\|\mathsf{L}_{\mathsf{Bag}}\|_2$, and no additional choice principle needs to be assumed in this case.

We conclude by discussing generalizations of our constructions to other analytic functors.

The material presented in the paper (apart from Section 7) has been formalized in the Cubical Agda proof assistant, building on top of the `agda/cubical` library [25]. The code is freely available at `https://github.com/phijor/agda-cubical-multiset`. For any result in the paper decorated with an `Identifier`, the repository contains instructions on how to find the corresponding formalization.

2 Type Theory and Cubical Agda

We work in homotopy type theory [26] and practically our formalization takes place in Cubical Agda [30]. We recall some basic notions that are employed in our development.

Given a type A and a type family B on A, the associated dependent function type is $(x : A) \to B\,x$, written also $\forall x.\, B\,x$ when the type A is clear from context. Implicit arguments of dependent functions are enclosed in curly brackets. Basic inductive types include: unit 1, empty $\bot$, naturals $\mathbb{N}$, finite prefixes of naturals $\mathsf{Fin}\,n$, lists $\mathsf{List}\,A$, dependent pair $\sum(x : A).\, B\,x$, binary sum $A + B$. We use standard names for their constructors. The unique function from a type A into the unit type is $! : A \to 1$. Given an inductive type T, we write elim_T and rec_T for its dependent and non-dependent elimination principles, respectively (we employ the same notation also for higher inductive types). The action on

maps of a functor $F : \mathsf{Type} \to \mathsf{Type}$ is map_F; to avoid ambiguities, we write $\mathsf{map}_F\, f$ over the conventional $F(f)$. Most of our constructions are universe-polymorphic, but for the sake of readability in the paper we use only the two lowest universe of types, Type and Type_1.

Given $x, y : A$, their definitional equality is denoted $x =_{\mathsf{df}} y$ while propositional equality is $x = y$. Following "cubical terminology", the latter is called the *path type* between x and y. In Cubical Agda, the path type $x = y$ behaves similarly to a function type $\mathbb{I} \to A$, where $\mathbb{I}$ is a primitive interval type with endpoints i_0 and i_1. An element $p : x = y$ is eliminated by application to an interval name $r : \mathbb{I}$, returning $p\, r : A$. But unlike function types, this application can compute even when p is unknown by using the endpoints x and y: $p\, i_0$ reduces to x and $p\, i_1$ reduces to y. Path introduction is lambda abstraction $(\lambda i : \mathbb{I}.\, t) : x = y$, but it causes the extra requirement to match the endpoints: $t[i_0/i]$ is judgementally equal to x and $t[i_1/i]$ is judgementally equal to y. We write $\mathsf{refl}\ x$ for the constant path (*i.e.* proof of reflexivity) in $x = x$ and $(\bullet)$ for sequential composition of paths.

A function $f : A \to B$ is an *equivalence* if it has contractible fibers, *i.e.* if the preimage of any element in B under f is a singleton type. Any function underlying a type isomorphism defines an equivalence. Writing $A \simeq B$ for the type of equivalences between A and B, Voevodsky's *univalence principle* states that the canonical function of type $A = B \to A \simeq B$ is an equivalence. This is a theorem in Cubical Agda. In particular, there is a function $\mathsf{ua} : A \simeq B \to A = B$ turning equivalences into path equalities. Univalence implies *function extensionality*: pointwise equal functions are equal. We recall the first instances of the hierarchy of *homotopy levels*,[1] and say that a type A is:

($n = 1$) a *proposition*, if $\mathsf{isProp}\, A =_{\mathsf{df}} (a\, b : A) \to a = b$ is inhabited,

($n = 2$) a *set*, if $\mathsf{isSet}\, A =_{\mathsf{df}} (a\, b : A) \to \mathsf{isProp}\ (a = b)$ is inhabited,

($n = 3$) a *groupoid*, if $\mathsf{isGroupoid}\, A =_{\mathsf{df}} (a\, b : A) \to \mathsf{isSet}\ (a = b)$ is inhabited.

When mentioning "sets" or "groupoids", we always refer to the definitions above.

A *higher inductive type* (HIT) is like an inductive type, but its constructors can build both its elements and its (higher) paths. HITs are primitively supported in Cubical Agda. We recall the definition of three basic HITs: propositional truncation, set-truncation and set-quotient.

The *propositional truncation* $\|A\|_1$ is the proposition associated to the type A, *i.e.* it identifies all the elements and (higher) paths of A. It is the HIT with constructors

$$\frac{a : A}{|a|_1 : \|A\|_1} \qquad \frac{x, y : \|A\|_1}{\mathsf{squash}_1\ x\ y : x = y}$$

We define the *existential quantifier* $\exists(x : A).\, B\, x =_{\mathsf{df}} \|\sum(x : A).\, B\, x\|_1$, which records the mere existence of an element x satisfying B.

The *set-truncation* $\|A\|_2$ is the set associated to the type A, *i.e.* it identifies all (higher) paths of A. It is the HIT with constructors

$$\frac{a : A}{|a|_2 : \|A\|_2} \qquad \frac{x, y : \|A\|_2 \qquad p, q : x = y}{\mathsf{squash}_2\ p\ q : p = q}$$

The *set-quotient* $A /_2 R$ of a type A by a (possibly proof-relevant) relation $R : A \to A \to \mathsf{Type}$ is the HIT with constructors

$$\frac{a : A}{[a]_2 : A /_2 R} \qquad \frac{a, b : A \qquad r : R\ a\ b}{\mathsf{eq}/_2\ r : [a]_2 = [b]_2} \qquad \frac{x, y : A /_2 R \qquad p, q : x = y}{\mathsf{squash}/_2\ p\ q : p = q}$$

[1] To stay close to the formalization, we follow Voevodsky's [31] 0-based numbering of h-levels.

The term $[a]_2$ is the R-equivalence class of a, while the path constructor $\mathsf{eq}/_2$ states that R-related elements have path equal equivalence classes. The higher path constructor $\mathsf{squash}/_2$ forces $A /_2 R$ to be a set.

Other HITs are presented in the next section, where we also take a closer look at their elimination principles.

3 The Finite Bag Functor in Sets

The action of the finite bag functor on a type X can be encoded as a higher inductive type in various ways, three of which are presented here. The first is the algebraic presentation of the free commutative monoid, the second as lists modulo permutations, the third as an analytic functor. These are all set-based definitions, in the sense that the type of finite bags is a set. In Section 3.4 we prove these are naturally equivalent as types, therefore being equivalent as functors. Groupoid-based definitions are discussed in Section 5.

3.1 As the Free Commutative Monoid

Given a type X, the *free commutative monoid* on X [8] is the HIT induced by the following rules:

$$\frac{}{\varepsilon : \mathsf{FCM}\, X} \qquad \frac{x : X}{\eta\, x : \mathsf{FCM}\, X} \qquad \frac{xs, ys : \mathsf{FCM}\, X}{xs \oplus ys : \mathsf{FCM}\, X}$$

$$\frac{xs : \mathsf{FCM}\, X}{\mathsf{unit} : \varepsilon \oplus xs = xs} \qquad \frac{xs, ys, zs : \mathsf{FCM}\, X}{\mathsf{assoc} : xs \oplus (ys \oplus zs) = (xs \oplus ys) \oplus zs}$$

$$\frac{xs, ys : \mathsf{FCM}\, X}{\mathsf{comm} : xs \oplus ys = ys \oplus xs} \qquad \frac{xs, ys : \mathsf{FCM}\, X \qquad p, q : xs = ys}{\mathsf{squash}_{\mathsf{FCM}}\, p\, q : p = q}$$

The constructor η embeds X into $\mathsf{FCM}\, X$, while ε and $\oplus$ are the unit and multiplication of the monoid. The path constructors express unitality of ε with respect to $\oplus$, associativity and commutativity of $\oplus$, and the final higher path constructor forces $\mathsf{FCM}\, X$ to be a set.

In Cubical Agda, functions out of HITs like $\mathsf{FCM}\, X$ can be defined directly by pattern matching. But it is often useful to have elimination principles at hand that give more control on the shape of the proof obligations. For example, the non-dependent elimination principle of $\mathsf{FCM}\, X$ states that a function of type $\mathsf{FCM}\, X \to A$ is definable, provided that A is a commutative monoid and there exists a function $\eta^* : X \to A$.

$$\begin{aligned}
&\mathsf{rec}_{\mathsf{FCM}\, X} : \{A : \mathsf{Type}\} \to \mathsf{isSet}\, A \\
&\qquad \to (\varepsilon^* : A)\, (\eta^* : X \to A)\, ((+) : A \to A \to A) \\
&\qquad \to (\forall a.\ \varepsilon^* + a = a) \\
&\qquad \to (\forall a\, b\, c.\ a + (b + c) = (a + b) + c) \\
&\qquad \to (\forall a\, b.\ a + b = b + a) \\
&\qquad \to \mathsf{FCM}\, X \to A
\end{aligned}$$

FCM is a functor, with action on maps given by

$$\begin{aligned}
&\mathsf{map}_{\mathsf{FCM}} : (f : X \to Y) \to \mathsf{FCM}\, X \to \mathsf{FCM}\, Y \\
&\mathsf{map}_{\mathsf{FCM}}\ f =_{\mathsf{df}} \mathsf{rec}_{\mathsf{FCM}\, X}\ \ \mathsf{squash}_{\mathsf{FCM}}\ \varepsilon\ (\eta \circ f)\ (\oplus)\ \mathsf{unit}\ \mathsf{assoc}\ \mathsf{comm}
\end{aligned}$$

3.2 As a Quotient of Lists

Another standard definition of the type of finite bags is as lists modulo permutations. The relation specifying the existence of a permutation between two lists can be given in multiple ways, here we mention two possibilities.

Given $xs, ys : \mathsf{List}\, X$, the relation $\mathsf{Perm}\, xs\, ys$ is generated by the rules:

$$\frac{}{\mathsf{Perm}\, xs\, xs} \qquad \frac{\mathsf{Perm}\, (xs +\!\!+ x :: y :: ys)\, zs}{\mathsf{Perm}\, (xs +\!\!+ y :: x :: ys)\, zs}$$

In other words, Perm is the reflexive-transitive closure of the relation generated by pairs of lists of the form $xs +\!\!+ x :: y :: ys$ and $xs +\!\!+ y :: x :: ys$. This is a very "intensional" way of representing permutations of lists: a proof of $\mathsf{Perm}\, xs\, ys$ not only records where each entry in xs is moved to in ys, but also how it is moved there. As such, $\mathsf{Perm}\, xs\, ys$ is generally not a proposition.

Another way of specifying permutations is via a *relation lifting*, often called a *relator* [19]. Given a relation R on a type X, we inductively define a relation $\mathsf{DRelator}\, R$ on $\mathsf{List}\, X$, which intuitively states that each occurrence of an element x in the first list is R-related to the occurrence of an element y in the second list. The type of occurrences $x \in xs$ is generated by

$$\frac{x : X \qquad xs : \mathsf{List}\, X}{x \in x :: xs} \qquad \frac{x\, y : X \qquad xs : \mathsf{List}\, X \qquad m : x \in xs}{x \in y :: xs}$$

Removal $xs \setminus m : \mathsf{List}\, X$ of an occurrence $m : x \in xs$ is defined by induction on m. The *directed relation lifting* of R is the relation generated by rules

$$\frac{}{\mathsf{DRelator}\, R\, [\,]\, ys} \qquad \frac{\exists(y : Y).\ \sum(m : y \in ys).\ R\, x\, y \times \mathsf{DRelator}\ R\, xs\, (ys \setminus m)}{\mathsf{DRelator}\, R\, (x :: xs)\ ys}$$

and we take the relation lifting of R to be the *symmetrization* of the relation $\mathsf{DRelator}\, R$, *i.e.* $\mathsf{Relator}\, R\ xs\ ys =_{\mathsf{df}} \mathsf{DRelator}\, R\ xs\ ys \times \mathsf{DRelator}\, R\ ys\ xs$. Because of the presence of a propositional truncation in the premise of the 2nd rule, both $\mathsf{DRelator}\, R$ and $\mathsf{Relator}\, R$ are propositionally-valued. If R is reflexive and transitive, then $\mathsf{Relator}\, R$ is an equivalence relation.

When R is path equality on X, the type $\mathsf{Relator}\, (=)\ xs\ ys$ expresses the mere existence of a permutation connecting xs and ys. In fact, $\|\mathsf{Perm}\ xs\ ys\|_1$ and $\mathsf{Relator}\, (=)\ xs\ ys$ are equivalent types.

3.3 As an Analytic Functor

We additionally introduce the type of finite bags over X as an *analytic functor* (in the formulation of Hasegawa [12]). For any type X, we define a type of *finite multisets* [20, 8]

$$\mathsf{FMSet}\, X =_{\mathsf{df}} \sum(n : \mathbb{N}).\ (\mathsf{Fin}\, n \to X)\,/_2\, \mathsf{SymAct}\, n$$

where $\mathsf{SymAct}\, n$ is the propositionally-valued relation

$$\mathsf{SymAct}\, n\ v\ w =_{\mathsf{df}} \exists(\sigma : \mathsf{Fin}\, n \simeq \mathsf{Fin}\, n).\ v = w \circ \sigma$$

In other words, an element of $\mathsf{FMSet}\, X$ is a pair of a natural number n (the size of the set) and an equivalence class of functions $v : \mathsf{Fin}\, n \to X$ picking an element in X for each index $k < n$. The relation $\mathsf{SymAct}\, n$ is the action of the symmetric group $\mathsf{Fin}\, n \simeq \mathsf{Fin}\, n$ on n-tuples of elements of X. We write $\mathsf{SymAct}_\infty\, n$ for the non-propositionally-truncated variant of $\mathsf{SymAct}\, n$. We write $v \sim w$ instead of $\mathsf{SymAct}\, n\ v\ w$ when n is clear from context, and analogously $(\mathsf{Fin}\, n \to X)\,/_2 \sim$ in place of $(\mathsf{Fin}\, n \to X)\,/_2\, \mathsf{SymAct}\, n$.

The proof of Theorem 23 employs the fact that FMSet is invariant under set-truncation. The latter fact factors through the following lemma, stating that set-truncation distributes over finite families of types.

▶ **Lemma 1** (finChoiceEquiv). *For any* $n : \mathbb{N}$ *and type family* $Y : \mathsf{Fin}\, n \to \mathsf{Type}$*, there is an equivalence* $\mathsf{box} : ((k : \mathsf{Fin}\, n) \to \|Y\, k\|_2) \simeq \|(k : \mathsf{Fin}\, n) \to Y\, k\|_2$.

Proof. We sketch a proof for a constant type family $Y = (\lambda\_.\ X)$. The dependent case is analogous. The function underlying the equivalence is defined by induction on n. For $n = 0$ we have $\mathsf{Fin}\, 0 \simeq \bot$, so $\mathsf{box} =_{\mathsf{df}} (\lambda\_.\ |\mathsf{elim}_{\bot}|_2)$. In the inductive step, we lift the derivable "cons" operation $(::) : X \to (\mathsf{Fin}\, n \to X) \to (\mathsf{Fin}\,(1 + n) \to X)$ to the set-truncation. A two-sided inverse $\mathsf{unbox} : \|\mathsf{Fin}\, n \to X\|_2 \to \mathsf{Fin}\, n \to \|X\|_2$ of box is given by $\mathsf{unbox}\, \bar{v}\, k =_{\mathsf{df}} \mathsf{map}_{\|\_\|_2}(\lambda v.\ v\, k)\, \bar{v}$. ◀

The equivalence of Lemma 1 allows to define a variant of the elimination principle $\mathsf{elim}_{\|X\|_2}$ taking $\mathsf{Fin}\, n \to \|X\|_2$ as input instead of $\|X\|_2$ (a sort of "finite choice" principle for set-truncation):

$$\begin{aligned} \mathsf{elim}_{\|X\|_2\mathsf{fin}} : \{n : \mathbb{N}\}\, \{B : (\mathsf{Fin}\, n \to \|X\|_2) \to \mathsf{Type}\} \{s_B : \forall v.\ \mathsf{isSet}(B\, v)\} \\ \to (c : (w : \mathsf{Fin}\, n \to X) \to B\, (|\_|_2 \circ w)) \\ \to (v : \mathsf{Fin}\, n \to \|X\|_2) \to B\, v \end{aligned} \tag{2}$$

This comes with a (propositional) computation rule $\mathsf{elim}^{\beta}_{\|X\|_2\mathsf{fin}} : \mathsf{elim}_{\|X\|_2\mathsf{fin}}\ c\ (|\_|_2 \circ v) = c\, v$.

▶ **Theorem 2** (FMSetTruncInvariance). FMSet *is invariant under set-truncation: for any type* X*, there is an equivalence* $\mathsf{FMSet}\, \|X\|_2 \simeq \mathsf{FMSet}\, X$.

Proof. The equivalence is obtained from an isomorphism. The right-to-left function is $\mathsf{map}_{\mathsf{FMSet}}\ |\_|_2$. For the left-to-right direction, we use $\mathsf{elim}_{\|X\|_2\mathsf{fin}}$ in (2) to define a function typed $(\mathsf{Fin}\, n \to \|X\|_2) \to (\mathsf{Fin}\, n \to X)\,/_2\, \sim$ that turns set-truncation into a set-quotient, which is enough to obtain a function typed $\mathsf{FMSet}\, \|X\|_2 \to \mathsf{FMSet}\, X$. That these maps are mutual inverses follows from $\mathsf{elim}^{\beta}_{\|X\|_2\mathsf{fin}}$. ◀

3.4 Equivalence of Presentations

All encodings of finite multisets used in the preceding section induce equivalent functors:

▶ **Proposition 3** (FMSetEquivs). *For any type* X*, there is a sequence of equivalences*

$$\mathsf{FCM}\, X \overset{\alpha}{\simeq} \mathsf{List}\, X\,/_2\, \mathsf{Perm} \overset{\beta}{\simeq} \mathsf{List}\, X\,/_2\, \mathsf{Relator}\ (=) \overset{\gamma}{\simeq} \mathsf{FMSet}\, X,$$

which are natural in X*: for any* $f : X \to Y$*,* $\alpha \circ \mathsf{map}_{\mathsf{FCM}}\, f = \mathsf{map}_{\mathsf{List}\, X\,/_2\, \mathsf{Relator}\ (=)}\, f \circ \alpha$*, and similarly for* β *and* γ*.*

Proof. Equivalence α is obtained by observing that both types form a free commutative monoid on X, with addition $(\oplus)$ and $(+\!+)$ respectively. For β, note that Relator $(=)$ is a propositionally-valued relation, while Perm is generally not. Yet it is enough to provide a bi-implication between the relations to conclude that the set-quotients they define are equivalent, as mentioned in Section 3.2. Equivalence γ is obtained similarly, this time proving that the encodings of permutations ("intensionally" via the relator and "extensional" in terms of equivalence of types) are logically equivalent. Naturality is established directly. ◀

In the formalization, we make use of slight variations of the above types where convenient. These mostly concern presentation of lists (*e.g.* bundling lengths via $\mathsf{List}\, A \simeq \sum_{n:\mathbb{N}} \mathsf{Vec}\, A\, n$), and are easily seen to be naturally equivalent.

3.5 Definable Quotients and Sorting

In the absence of the axiom of choice, it is not generally possible to define a section of the equivalence class constructor $[\_]_2 : A \to A /_2 R$. A set-quotient $A /_2 R$ for which such a section exists is called *definable* [20]. Spelled out, there is a representative-picking function $\mathsf{rep} : A /_2 R \to A$ such that $[\mathsf{rep}\, x]_2 = x$ for all $x : A /_2 R$.

In the proof of Theorem 11 we employ the fact that the type of finite bags $\mathsf{FMSet}\, X$, for some specific choice of X, is linearly-ordered and $(\mathsf{Fin}\, n \to X) /_2 \sim$ is a definable set-quotient. A relation $(<)$ is a *linear order* when it is asymmetric, transitive, propositionally-valued and total, in the sense that the trichotomy $(x < y) + (x = y) + (y < x)$ holds for all $x, y : X$. If X is a set with linear order $(<)$, then lists over X can be sorted with respect to $(<)$ via a function $\mathsf{sort} : \mathsf{List}\, X \to \mathsf{List}\, X$ essentially implementing the insertion-sort algorithm, which allows the construction of a permutation typed $\mathsf{Perm}\ xs\ (\mathsf{sort}\, xs)$. Sorting is independent of the positions of each entry in the input list, therefore via $\mathsf{rec}_{\mathsf{List}\, X /_2 \mathsf{Perm}}$ we obtain a function $\mathsf{sortPerm} : \mathsf{List}\, X /_2 \mathsf{Perm} \to \mathsf{List}\, X$. It is not hard to show that $\mathsf{sortPerm}$ is a section of the equivalence class constructor, so $\mathsf{List}\, X /_2 \mathsf{Perm}$ is a definable quotient. Since $\mathsf{List}\, X /_2 \mathsf{Perm} \simeq \mathsf{FMSet}\, X$, we obtain the following result.

▶ **Proposition 4** (`SymActDefinable`). *If X is a linearly-ordered set, then $(\mathsf{Fin}\, n \to X) /_2 \sim$ is a definable quotient for all $n : \mathbb{N}$.*

In the presence of a linear order $(<)$ on X, we can extract from any proof that two lists are merely related by a permutation an actual permutation witnessing this:

▶ **Proposition 5** (`SymActUntruncate`). *If X is a linearly-ordered set, then for all $n : \mathbb{N}$ and $v, w : \mathsf{Fin}\, n \to X$ there exists a function typed $\mathsf{SymAct}\, n\ v\ w \to \mathsf{SymAct}_\infty\, n\ v\ w$,* i.e. *the propositional truncation in $\mathsf{SymAct}\, n\ v\ w$ can be removed.*

Proof. Since $\mathsf{FMSet}\, X \simeq \mathsf{List}\, X /_2 \mathsf{Perm}$, it is enough to define for all $xs\ ys : \mathsf{List}\, X$ a function $\|\mathsf{Perm}\ xs\ ys\|_1 \to \mathsf{Perm}\ xs\ ys$. To escape the truncation, we first implement a function $\mathsf{canonPerm} : \mathsf{Perm}\ xs\ ys \to \mathsf{Perm}\ xs\ ys$ returning a "canonical" way of permuting xs into ys. Given $\sigma : \mathsf{Perm}\ xs\ ys$, sorting yields a path $p_\sigma : \mathsf{sort}\, xs = \mathsf{sort}\, ys$. Composing (along p_σ) the permutations obtained from sorting xs and (un-)sorting ys gives the desired term $\mathsf{canonPerm}\, \sigma$. Since X is a set, p_σ lands in a proposition. Thus, $\mathsf{canonPerm}$ is *weakly constant* and lifts to a function from the truncation [7, Corollary 2]. ◀

To illustrate the computational behavior of `canonPerm`, see `BraidExample` in the code.

The order $(<)$ can be extended to a linear order on $\mathsf{List}\, X$ via the *lexicographic order*

$$\frac{}{\mathsf{Lex}\,(<)\ [\,]\ (y :: ys)} \qquad \frac{x < y}{\mathsf{Lex}\,(<)\ (x :: xs)\ (y :: ys)} \qquad \frac{x = y \qquad \mathsf{Lex}\,(<)\ xs\ ys}{\mathsf{Lex}\,(<)\ (x :: xs)\ (y :: ys)}$$

and further to $\mathsf{List}\, X /_2 \mathsf{Perm}$ by defining $\mathsf{LexPerm}\,(<)\ x\ y =_{\mathsf{df}} \mathsf{Lex}\,(<)\ (\mathsf{sortPerm}\, x)\ (\mathsf{sortPerm}\, y)$.

▶ **Proposition 6** (`linLexFMSet`). *If X is a linearly-ordered set, then $(\mathsf{Fin}\, n \to X) /_2 \sim$ is linearly-ordered for all $n : \mathbb{N}$.*

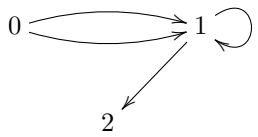
4 The Final Coalgebra in Sets

We now turn to constructing the final coalgebra of the finite bag functor, given by one of the equivalent definitions in Section 3.

Given a functor $F : \mathsf{Type} \to \mathsf{Type}$, the types of *coalgebras* and *coalgebra morphisms* between two coalgebras (A, a) and (B, b) are

$$\mathsf{Coalg}\ F =_{\mathsf{df}} \sum (A : \mathsf{Type}).\ A \to FA$$
$$\mathsf{CoalgMor}\ F\ (A, a)\ (B, b) =_{\mathsf{df}} \sum (f : A \to B).\ b \circ f = (\mathsf{map}_F\ f) \circ a$$

Coalgebras can be used to represent transition systems. For example, the coalgebra on the right encodes the small transition system on the left:



$$c : \mathsf{Fin}\,3 \to \mathsf{FCM}(\mathsf{Fin}\,3)$$
$$c\ 0 =_{\mathsf{df}} \eta\ 1 \oplus \eta\ 1$$
$$c\ 1 =_{\mathsf{df}} \eta\ 1 \oplus \eta\ 2$$
$$c\ 2 =_{\mathsf{df}} \varepsilon$$

A coalgebra is *final* if there exists a unique coalgebra morphism from any other coalgebra. This can be formalized by saying that there is a coalgebra $C : \mathsf{Coalg}\ F$ such that the type $\mathsf{CoalgMor}\ F\ D\ C$ is contractible for any other coalgebra D. These definitions are the same of Ahrens *et al.* [3], which they only consider in the case of F being a polynomial functor.

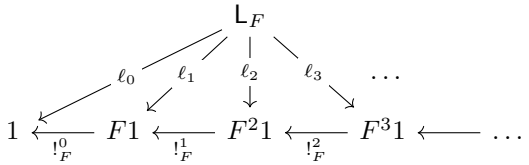
We analyze two constructions of the final coalgebra for the finite bag functor: as an ω-limit and as a set-quotient of the final coalgebra of the List functor.

4.1 As an ω-Limit

Consider the chain in (1), for some $F : \mathsf{Type} \to \mathsf{Type}$. We formally define $F^n 1$ by recursion on n: $F^0 1 =_{\mathsf{df}} 1$ and $F^{1+n} 1 =_{\mathsf{df}} F(F^n 1)$. Similarly we can define the iteration $\mathsf{map}^n_F\ !$, which we denote $!^n_F$. In HoTT, the *(homotopy) limit* of the chain is definable as

$$\lim_n (F^n 1) =_{\mathsf{df}} \sum (x : (n : \mathbb{N}) \to F^n 1).\ \forall n.\ !^n_F\ (x\ (1 + n)) = x\ n$$

Write $\mathsf{L}_F =_{\mathsf{df}} \lim_n (F^n 1)$. An element of the limit consists of an element $x\,n : F^n 1$, for all $n : \mathbb{N}$, and a proof that restricting $x\,(1 + n)$ to $F^n 1$ via $!^n_F$ is equal to $x\,n$. Writing $\ell_n : \mathsf{L}_F \to F^n 1$ for the n-th projection from the limit, we obtain the usual diagram:



(3)

The limit is invariant with respect to shifting the chain by one position, *i.e.* there is an equivalence $\mathsf{shift} : \mathsf{L}_F \simeq \mathsf{L}^{\mathsf{sh}}_F$, where $\mathsf{L}^{\mathsf{sh}}_F =_{\mathsf{df}} \lim_n (F^{1+n} 1)$. We use $\ell_n : \mathsf{L}^{\mathsf{sh}}_F \to F^{1+n} 1$ also for the n-th projection from the shifted limit. If F is set-valued, the limit L_F is also a set. Notice that for naturally equivalent and set-valued F and G, we have $\mathsf{L}_F \simeq \mathsf{L}_G$, since naturally equivalent chains have equivalent limits; see `chainEquivToLimitEquiv` in the formalization for details. This implies that, when proving properties of this limit for finite bags, we can use any of the equivalent presentations in Proposition 3 as convenient.

In classical set theory, $\mathsf{L}_{\mathsf{FMSet}}$ can be proved to be the final coalgebra of FMSet. The proof proceeds by first constructing a function $\mathsf{pres}_{\mathsf{FMSet}} : \mathsf{FMSet}\ \mathsf{L}_{\mathsf{FMSet}} \to \mathsf{L}^{\mathsf{sh}}_{\mathsf{FMSet}}$ via the universal property of the limit: take $\ell_n\ (\mathsf{pres}_{\mathsf{FMSet}}\ s)$ as $\mathsf{map}_{\mathsf{FMSet}}\ \ell_n\ s$. The function $\mathsf{pres}_{\mathsf{FMSet}}$ is then proved to be an equivalence, showing that FMSet preserves the ω-limit. The composition of shift with the inverse $\mathsf{pres}^{-1}_{\mathsf{FMSet}}$ provides a coalgebra structure for $\mathsf{L}_{\mathsf{FMSet}}$. This can be proved to be final, again using the universal property of the limit.

Constructively, there are issues in proving that $\mathsf{pres}_{\mathsf{FMSet}}$ is an equivalence. Its injectivity is equivalent to the *lesser limited principle of omniscience* (LLPO) [6, Ch. 1]. The latter is a weak version of the law of the excluded middle, and it is not provable from intuitionistic axioms alone. It states that, given an infinite stream of Boolean values that yields true in at most one position, one can decide whether all even or all odd positions are false. Both injectivity of $\mathsf{pres}_{\mathsf{FMSet}}$ and LLPO are propositions, so to establish an equivalence, it is sufficient to find a bi-implication between them.

▶ **Theorem 7** (`InjectiveFMSetPresToLLPO`). *If* $\mathsf{pres}_{\mathsf{FMSet}}$ *is injective, then LLPO holds.*

Proof. In this proof we use FCM instead of FMSet. The statement of the theorem holds since $\mathsf{FCM}\,X$ is naturally equivalent to $\mathsf{FMSet}\,X$. It is sufficient to show that the injectivity of $\mathsf{pres}_{\mathsf{FCM}}$ implies that the following type is inhabited:

$$\begin{aligned}&(x, y_1, y_2 : \mathsf{L}_{\mathsf{FCM}}) \to (ys : \mathbb{N} \to \mathsf{L}_{\mathsf{FCM}})\\ &\quad \to (\mathsf{split} : \forall n.\ ys\ n = y_1 + ys\ n = y_2)\ (\mathsf{diag} : \forall n.\ \ell_n\ x = \ell_n\ (ys\ n))\\ &\quad \to \|x = y_1 + x = y_2\|_1\end{aligned} \tag{4}$$

This is a form of *completeness* of two-element subsets of the ω-limit: every converging sequence ys consisting of elements from the subset $\{y_1, y_2\}$ has its limit x also belonging to the subset $\{y_1, y_2\}$. Mandelkern [21] has proved the equivalence of LLPO with completeness of two-element subsets of real numbers. We adapt their proof, for details refer to either [29, Theorem 7] or the formalization (`CompleteToLLPO`).

To prove completeness, assume $x, y_1, y_2, ys, \mathsf{split}$ and diag as in (4). Using split, define the complement of ys as $\overline{ys}\ n =_{\mathsf{df}} y_2$ if $ys\ n = y_1$ and $\overline{ys}\ n =_{\mathsf{df}} y_1$ if $ys\ n = y_2$. The diagonal of $\overline{ys}$ also has the limit-property, *i.e.*

$$\forall n.\ !^n_{\mathsf{FCM}}(\ell_{1+n}\ (\overline{ys}\ (1+n))) = \ell_n\ (\overline{ys}\ n) \tag{5}$$

For this, fix n and check the four cases generated by inspecting $\mathsf{split}\,n$ and $\mathsf{split}\,(1+n)$. In one case, (5) reduces to the limit-property of y_1, in another to that of y_2 and in the remaining cases to Lemma 8 below. Call $\overline{x} : \mathsf{L}_{\mathsf{FCM}}$ the element of the limit such that $\ell_n\,\overline{x} =_{\mathsf{df}} \ell_n\,(\overline{ys}\ n)$.

Write $\{x, y\} =_{\mathsf{df}} \eta\,x \oplus \eta\,y$ for the two-element bag comprising of x and y. For all n, we know that $\{ys\ n, \overline{ys}\ n\} = \{y_1, y_2\}$ either by refl or $\oplus\mathsf{comm}$, depending on $\mathsf{split}\,n$. Using the latter equality, the definition of $\mathsf{pres}_{\mathsf{FCM}}$ and the assumption diag, we can form the following sequence of equalities:

$$\begin{aligned}\ell_n\,(\mathsf{pres}_{\mathsf{FCM}}\{x, \overline{x}\}) &= \{\ell_n\,x, \ell_n\,\overline{x}\} = \{\ell_n\,(ys\ n), \ell_n\,(\overline{ys}\ n)\}\\ &= \ell_n\,(\mathsf{pres}_{\mathsf{FCM}}\{ys\ n, \overline{ys}\ n\}) = \ell_n\,(\mathsf{pres}_{\mathsf{FCM}}\{y_1, y_2\})\end{aligned}$$

which implies $\mathsf{pres}_{\mathsf{FCM}}\{x, \overline{x}\} = \mathsf{pres}_{\mathsf{FCM}}\{y_1, y_2\}$. From the injectivity of $\mathsf{pres}_{\mathsf{FCM}}$ it follows that $\{x, \overline{x}\} = \{y_1, y_2\}$, which also implies that (merely) $x = y_1$ or $x = y_2$. ◀

The above depends on a property of sequences in L_F that are "approximated" by some $x : \mathsf{L}_F$:

▶ **Lemma 8** (`LimitAlternationLemma`). *For any functor* F *where* $x : \mathsf{L}_F$ *and* $ys : \mathbb{N} \to \mathsf{L}_F$ *such that* $P : \forall n.\ \ell_n\,x = \ell_n\,ys_n$*, we have* $\forall n.\ !^n_F(\ell_{n+1}\,ys_n) = \ell_n\,ys_{n+1}$.

Proof. By alternating application of the limit property and assumption P, we obtain

$$!^n_F(\ell_{n+1}\,ys_n) = \ell_n\,ys_n \overset{P_n}{=} \ell_n\,x = !^n_F(\ell_{n+1}\,x) \overset{P_{n+1}}{=} !^n_F(\ell_{n+1}\,ys_{n+1}) = \ell_n\,ys_{n+1}$$ ◀

▶ **Theorem 9** (LLPOToInjectiveFMSetPres). *LLPO implies the injectivity of* $\mathsf{pres}_{\mathsf{FMSet}}$.

For the proof of Theorem 9, which employs the functor $\mathsf{List}(-)\,/_2\,\mathsf{Relator}\,(=)$ instead of FMSet, we refer the reader to our Agda formalization. The proof is similar to the one of a related result [29, Theorem 9]: the injectivity of $\mathsf{pres}_{\mathsf{Pfin}} : \mathsf{Pfin}\ \mathsf{L}_{\mathsf{Pfin}} \to \mathsf{L}^{\mathsf{sh}}_{\mathsf{Pfin}}$, where Pfin is the finite powerset functor, is derivable from LLPO and the axiom of countable choice. It turns out that countable choice is not needed, neither in Theorem 9 nor in Theorem 9 of [29].

Nevertheless, we are able to salvage the fact that $\mathsf{pres}_{\mathsf{FMSet}}$ has a section/right-inverse which targets the shifted limit. This implies that FMSet weakly preserves the ω-limit $\mathsf{L}_{\mathsf{FMSet}}$, but strong limit-preservation is equivalent to LLPO.

▶ **Lemma 10** (linLexIterFMSet). *For all* n, $\mathsf{FMSet}^n\,1$ *is linearly-ordered.*

Proof. Define $(<^n) : \mathsf{FMSet}^n\,1 \to \mathsf{FMSet}^n\,1 \to \mathsf{Type}$ by induction on n: $(<^0)$ is the empty relation and $x <^{1+n} y =_{\mathsf{df}} \mathsf{LexPerm}\,(<^n)\,x\,y$. Since the empty relation is linear, Proposition 6 implies that the order $(<^n)$ is linear for all $n : \mathbb{N}$. ◀

▶ **Theorem 11** (FMSetPresSection). *The function* $\mathsf{pres}_{\mathsf{FMSet}}$ *has a section.*

Proof. Let $s : \mathsf{L}^{\mathsf{sh}}_{\mathsf{FMSet}}$, we build an element $t : \mathsf{FMSet}\ \mathsf{L}_{\mathsf{FMSet}}$ in the fiber of $\mathsf{pres}_{\mathsf{FMSet}}$ over s. The size (*i.e.* the 1st projection) of the bags $\ell_n\,s$ is the same for all n, call it n^*. We set the size of t to be n^*. Given an index $k : \mathsf{Fin}\,n^*$, we now search for an element $u\,k : \mathsf{L}_{\mathsf{FMSet}}$ for defining $t =_{\mathsf{df}} (n^*, u)$.

For each $d : \mathbb{N}$, we know that $\ell_{d+1}\,s$ is path equal to a pair of the form (n^*, v_d). In order to construct u we need access to a representative of the equivalence class $v_d : (\mathsf{Fin}\,n^* \to \mathsf{FMSet}^d\,1)\,/_2\,\sim$ for each d. We know that this can be done using Lemma 10 and Proposition 4. Let $w_d : \mathsf{Fin}\,n^* \to \mathsf{FMSet}^d\,1$ be the canonical representative of v_d. The limit-property of s can be translated to the mere existence of a permutation $\sigma_d : \mathsf{Fin}\,n^* \simeq \mathsf{Fin}\,n^*$ such that $p : !^d_{\mathsf{FMSet}}(w_{1+d}\,k) = w_d\,(\sigma_d\,k)$, for all $d : \mathbb{N}$ and $k : \mathsf{Fin}\,n^*$. The construction of u also requires access to each permutation σ_d, which sits inside a propositional truncation for each d. We can access all these permutations by invoking Lemma 10 and Proposition 5.

We now have all the ingredients for building u. Define a permutation $\sigma^*_d : \mathsf{Fin}\,n^* \simeq \mathsf{Fin}\,n^*$ by induction on d: $\sigma^*_0 =_{\mathsf{df}} \mathsf{id}$, $\sigma^*_{1+d} =_{\mathsf{df}} \sigma_d^{-1} \circ \sigma^*_d$. Then take u such that $\ell_d\,(u\,k) =_{\mathsf{df}} w_d\,(\sigma^*_d\,k)$. One can show that $u\,k : \mathsf{L}_{\mathsf{FMSet}}$ for all $k : \mathsf{Fin}\,n^*$ since $!^d_{\mathsf{FMSet}}(w_{d+1}(\sigma^*_{d+1}\,k)) \overset{p}{=} w_d(\sigma_d(\sigma^*_{d+1}\,k)) = w_d(\sigma_d(\sigma_d^{-1}(\sigma^*_d\,k))) = w_d(\sigma^*_d\,k)$, and that t is indeed in the fiber of $\mathsf{pres}_{\mathsf{FMSet}}$ over s. ◀

By *Lambek's theorem*, every final coalgebra is necessarily an equivalence. Assuming LLPO we have $\mathsf{FMSet}\,\mathsf{L}_{\mathsf{FMSet}} \simeq \mathsf{L}_{\mathsf{FMSet}}$, and proving that the coalgebra underlying this equivalence is final in the category of sets is straightforward using the universal property of the limit.

4.2 As a Quotient of the Final List-Coalgebra

Instead of considering a type of unordered trees quotiented at each step of the construction, we investigate whether it is possible to define a final FMSet-coalgebra by quotienting the type of ordered trees by some suitable relation. It is known that the limit $\mathsf{L}_{\mathsf{List}} =_{\mathsf{df}} \lim_n(\mathsf{List}^n\,1)$ is the final coalgebra of the list functor in HoTT [3]. The limit $\mathsf{L}_{\mathsf{List}}$ is a type of non-wellfounded *ordered* trees, and we denote by $\mathsf{coalg}_{\mathsf{List}}$ its coalgebra structure. By choosing a suitable relation R, one can hope to obtain a type of *unordered* trees $\mathsf{L}_{\mathsf{List}}\,/_2\,R$ endowed with a FMSet-coalgebra structure. We choose R to be a notion of *bisimilarity* Bisim, obtained iteratively

from the *relation lifting* Relator applied to finite approximations of trees in $\mathsf{L}_{\mathsf{List}}$ [13]:

$$\begin{aligned}&\mathsf{Approx}^n : \mathsf{List}^n\ 1 \to \mathsf{List}^n\ 1 \to \mathsf{Type}\\&\mathsf{Approx}^0\ x\ y =_{\mathsf{df}} 1\\&\mathsf{Approx}^{1+n}\ x\ y =_{\mathsf{df}} \mathsf{Relator}\ (\mathsf{Approx}^n)\ x\ y.\end{aligned}$$

From the fact that $(\forall x, y.\ R\,x\,y \to S\,x\,y)$ implies $\forall xs, ys.\ \mathsf{Relator}\ R\ xs\ ys \to \mathsf{Relator}\ S\ xs\ ys$, we obtain, for $s, t : \mathsf{L}_{\mathsf{List}}$, a chain of propositions

$$\mathsf{Approx}^0(\ell_0\ s)\ (\ell_0\ t) \leftarrow \mathsf{Approx}^1(\ell_1\ s)\ (\ell_1\ t) \leftarrow \mathsf{Approx}^2(\ell_2\ s)\ (\ell_2\ t) \leftarrow \cdots \tag{6}$$

The desired relation $\mathsf{Bisim}\,s\,t$ is the limit of the chain in (6).

To find a coalgebra structure on $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$, we investigate whether $\mathsf{coalg}_{\mathsf{List}}$ lifts to a coalgebra of setoids $(\mathsf{L}_{\mathsf{List}}, \mathsf{Bisim}) \to (\mathsf{List}\,\mathsf{L}_{\mathsf{List}}, \mathsf{Relator}\,\mathsf{Bisim})$ and if so, whether this induces a (final) coalgebra on the quotient. For this, one needs to show that it is a *setoid-morphism*, *i.e.* for $s, t : \mathsf{L}_{\mathsf{List}}$, if $\mathsf{Bisim}\ s\ t$ then $\mathsf{Relator}\,\mathsf{Bisim}\ (\mathsf{coalg}_{\mathsf{List}}\ s)\ (\mathsf{coalg}_{\mathsf{List}}\ t)$. Once again, the same issue we found when trying to prove the injectivity of $\mathsf{pres}_{\mathsf{FMSet}}$ in Section 4.1 arises:

▶ **Theorem 12** (`isSetoidMorphismCoalgListToLLPO`). *If* $\mathsf{coalg}_{\mathsf{List}}$ *is a setoid-morphism, then LLPO holds.*

The proof is similar to that of Theorem 7. Similarly to Theorem 9, the converse is also true. Nevertheless, the inverse of $\mathsf{coalg}_{\mathsf{List}}$ is always a setoid-morphism. Therefore $\mathsf{coalg}_{\mathsf{List}}$ is an equivalence of setoids whenever it is a setoid-morphism, *i.e.* LLPO holds. Under this assumption alone it is the final coalgebra of an endofunctor in the category of setoids:

▶ **Theorem 13** (`finalFMSetoidCoalgebra`). *Assuming* $\mathsf{coalg}_{\mathsf{List}}$ *is a setoid-morphism, the setoid* $(\mathsf{L}_{\mathsf{List}}, \mathsf{Bisim})$ *has a coalgebra structure for the functor* $(X, R) \mapsto (\mathsf{List}\,X, \mathsf{Relator}\,R)$*, which is final in the category of setoids.*

Promisingly, we can show that the resulting quotient is a fixpoint for FMSet, and in particular a coalgebra (of sets):

▶ **Theorem 14** (`FMSetFixpointTree/Bisim`). *If* $\mathsf{coalg}_{\mathsf{List}}$ *is a setoid-morphism, it lifts to an equivalence* $\mathsf{coalg}_{\mathsf{FMSet}} : \mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim} \xrightarrow{\simeq} \mathsf{FMSet}(\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim})$.

Proof. For the proof, we employ the equivalent functor $\mathsf{List}(-) /_2 \mathsf{Relator}\ (=)$ instead of FMSet. The assumption implies that $\mathsf{coalg}_{\mathsf{List}}$ lifts to a function $\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim} \to \mathsf{FMSet}(\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim})$, definable by recursion on the set-quotient. An inverse $\mathsf{FMSet}(\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}) \to \mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$ is definable since $\mathsf{coalg}^{-1}_{\mathsf{List}}$ is always a setoid-morphism and $\mathsf{List}\,X /_2 \mathsf{Relator}\,R$ is an *effective quotient* for any setoid (X, R). ◀

However, like in case of the finite powerset, this fixpoint is not obviously the final FMSet-coalgebra. We were able to prove this assuming the axiom of choice:

▶ **Theorem 15** (`FinalFMSetCoalgebra`). *Assuming the axiom of choice, the fixpoint of Theorem 14 is the final* FMSet*-coalgebra in the category of sets.*

Proof. Define abbreviations $U =_{\mathsf{df}} \mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}$ and $R =_{\mathsf{df}} \mathsf{Relator}\ (=)$, and use the presentation of FMSet used in the proof of Theorem 14. To build a coalgebra morphism $u_c : C \to U$ from a given coalgebra $c : C \to \mathsf{FMSet}\,C$ to $\mathsf{coalg}_{\mathsf{FMSet}} : U \to \mathsf{FMSet}\,U$, one defines a function $u' : (C \to \mathsf{List}\,C) /_2 R^* \to (C \to U)$. Here, R^* is the pointwise lifting of R, and u' is obtained by recursion from the unique List-coalgebra morphism typed $C \to \mathsf{L}_{\mathsf{List}}$. The axiom of choice implies that the canonical map $(C \to \mathsf{List} /_2 R) \to (C \to \mathsf{List}\,C) /_2 R^*$ has a section θ for *arbitrary* C. This is sufficient to prove that $u_c =_{\mathsf{df}} u'(\theta(c))$ is the unique FMSet-coalgebra morphism from (C, c) to $(\mathsf{L}_{\mathsf{List}} /_2 \mathsf{Bisim}, \mathsf{coalg}_{\mathsf{FMSet}})$. ◀

5 The Finite Bag Functor in Groupoids

The results of Section 4 are evidence that the set-based definitions of finite bags from Section 3 are not fit for a fully constructive construction of the final coalgebra. In this section we study a groupoid-based definition and, following the ideas of Kock [16] and Finster *et al.* [10], argue that the correct perspective on finite bags in HoTT is to define them as groupoids instead of sets, particularly for the goal of final semantics. The rationale is that identifications of bags are permutations, and these should inherently be treated as *data*. Instead of viewing bags as quotients of lists, thereby "forgetting" about the permutations, we define a type of lists with "more identifications". Since all constructions based on this type have to be homotopy coherent, they will automatically respect the extra data, making them invariant under permutation for free. We define two equivalent type families Tote and Bag of finite bags valued in groupoids, and substantiate the previous claims by showing that the set-truncation of the former is equivalent to FMSet (Theorem 17), and constructing the final coalgebra of the latter in a straightforward way (Theorem 21 and Corollary 22).

First, recall one way of defining finite sets in HoTT [11]. A type B is called *(Bishop-)finite* if $\mathsf{isFinSet}\, B =_{\mathsf{df}} \sum(n : \mathbb{N}).\ \|B \simeq \mathsf{Fin}\, n\|_1$ holds, and we denote the collection of such types by $\mathsf{FinSet} =_{\mathsf{df}} \sum(B : \mathsf{Type}).\ \mathsf{isFinSet}\, B$. The underlying type of a FinSet is accessed via the first projection $\langle - \rangle : \mathsf{FinSet} \to \mathsf{Type}$.

The type $\mathsf{isFinSet}\, B$ is a proposition and any type B satisfying the predicate is a set. It follows that FinSet forms a groupoid. Note that FinSet is a *large* type, *i.e.* $\mathsf{FinSet} : \mathsf{Type}_1$. From this, we can define a "tote" (in the sense of a "large bag") $\mathsf{Tote} : \mathsf{Type} \to \mathsf{Type}_1$ as

$$\mathsf{Tote}\, X =_{\mathrm{df}} \sum (B : \mathsf{FinSet}).\ \langle B \rangle \to X,$$

Elements of $\mathsf{Tote}\, X$ are pairs consisting of a finite set B and a function from (the type underlying) B to X which picks the elements in the tote. Univalence implies that the path type $(B, v) = (C, w)$ in $\mathsf{Tote}\, X$ is equivalent to the type of dependent pairs consisting of an equivalence $\sigma : \langle B \rangle \simeq \langle C \rangle$ and a path $v = w \circ \sigma$. This indicates that $\mathsf{Tote}\, X$ is not a set, in general it is at least a groupoid.

▶ **Proposition 16** (`isGroupoidTote`). *If X is a groupoid, then $\mathsf{Tote}\, X$ is a groupoid.*

Proof. Since X is a groupoid, the function type $\langle B \rangle \to X$ is a groupoid for any $B : \mathsf{FinSet}$. The type FinSet is also a groupoid, so the entire Σ-type is a groupoid. ◀

Similar to how $\mathsf{FMSet}\, X$ is the free commutative monoid on X, $\mathsf{Tote}\, X$ can be proved equivalent to the *free symmetric monoidal groupoid* on X [22, Corollary 5.103], which serves as an alternative proof of MacLane's coherence for symmetric monoidal categories. It differs from $\mathsf{FMSet}\, X$ in that path equality in the former records the permutations between the (finite sets representing) sizes of the bags, while the second only cares about the mere existence of a permutation. Nonetheless, the two definitions become equivalent when we set-truncate the type of totes.

▶ **Theorem 17** (`FMSetToteTruncEquiv`, `isNatural-FMSetToteTruncEquiv`). *For any type X, $\|\mathsf{Tote}\, X\|_2 \simeq \mathsf{FMSet}\, X$. The equivalence is natural in X.*

Proof. The proof proceeds by constructing an isomorphism $\mathsf{FMSet}\, X \cong \|\mathsf{Tote}\, X\|_2$.

A function $\mathsf{toTote} : \mathsf{FMSet}\, X \to \|\mathsf{Tote}\, X\|_2$ is defined by first giving a function $f : \forall\{n\}.\ (\mathsf{Fin}\, n \to X) \to \|\mathsf{Tote}\, X\|_2$ and then showing that it respects $(\sim)$. Take $f(v) =_{\mathsf{df}} |(\mathsf{Fin}\, n, v)|_2$, since $\mathsf{Fin}\, n$ is a finite set. To prove that $v \sim w$ implies $f(v) = f(w)$, note that

the conclusion is a proposition, thus by invoking the recursion principle of propositional truncation we can assume given a permutation σ such that $r : v = w \circ \sigma$. By univalence, $\mathsf{ua}\,\sigma : \mathsf{Fin}\,n = \mathsf{Fin}\,n$, and transporting r along this path yields $p : (\mathsf{Fin}\,n, v) = (\mathsf{Fin}\,n, w)$. Then $\mathsf{cong}\,|\_|_2\,p : f(v) = f(w)$ as desired.

A function $\mathsf{toFMSet} : \|\mathsf{Tote}\,X\|_2 \to \mathsf{FMSet}\,X$ is defined via $\mathsf{rec}_{\|\mathsf{Tote}\,X\|_2}$, so it is enough to provide $g : \mathsf{Tote}\,X \to \mathsf{FMSet}\,X$. Assume given a finite set B of size n with $e : \|B \simeq \mathsf{Fin}\,n\|_1$ and $v : B \to X$. We would like to return something in $(\mathsf{Fin}\,n \to X)/_2 \sim$ by recursion on e, but this cannot work since the return type is a set. We can however employ a different recursion principle of propositional truncation [7, Corollary 2], which allows to define a function into a set provided that it is *(weakly) constant* (in the sense of [17]). Define $g' : (B \simeq \mathsf{Fin}\,n) \to (\mathsf{Fin}\,n \to X)/_2 \sim$ as $g'\,\alpha =_{\mathsf{df}} [v \circ \alpha]_2$, which can be proved to be constant and therefore well-defined. We can then take $g\,((B, n, e), v) =_{\mathsf{df}} (n, g'\,e)$. Proving $\mathsf{toFMSet} \circ \mathsf{toTote} = \mathsf{id}$ is straightforward. Proving $\mathsf{toTote} \circ \mathsf{toFMSet} = \mathsf{id}$ reduces to showing that $v \circ \alpha \sim v$ for any $v : B \to X$ and $\alpha : \mathsf{Fin}\,n \simeq B$, which is also direct. ◀

Before studying $\mathsf{L}_{\mathsf{Tote}}$, notice that the iteration $\mathsf{Tote}^n\,1$ is not well-typed as Tote targets a large universe. We could in principle define $\mathsf{Tote}' : \mathsf{Type}_1 \to \mathsf{Type}_1$ which does not raise the universe level by first lifting the unit type 1 to the universe Type_1. The resulting limit would be a large groupoid in $\mathsf{L}_{\mathsf{Tote}'} : \mathsf{Type}_1$. Instead, we define an equivalent small variant of Tote with the help of HITs.

Following [10], we first introduce an equivalent but small definition Bij of the type of finite sets FinSet. This is equivalent to the *groupoid-quotient* [24, 28] of the (categorical) groupoid with objects given by natural numbers and morphisms between n and m given by equivalences in $\mathsf{Fin}\,m \simeq \mathsf{Fin}\,n$. It is possible to prove that hom also preserves identities and inverses.

$$\frac{n : \mathbb{N}}{\mathsf{obj}\,n : \mathsf{Bij}} \qquad \frac{m, n : \mathbb{N} \qquad \alpha : \mathsf{Fin}\,m \simeq \mathsf{Fin}\,n}{\mathsf{hom}\;\alpha : \mathsf{obj}\,m = \mathsf{obj}\,n}$$

$$\frac{m, n, o : \mathbb{N} \qquad \alpha : \mathsf{Fin}\,m \simeq \mathsf{Fin}\,n \qquad \beta : \mathsf{Fin}\,n \simeq \mathsf{Fin}\,o}{\mathsf{hom}\,(\beta \circ \alpha) = \mathsf{hom}\;\alpha \bullet \mathsf{hom}\;\beta} \qquad \frac{}{\mathsf{isGroupoid}\;\mathsf{Bij}}$$

▶ **Proposition 18** (`BinFinSetEquiv`). *There is an equivalence* $\mathsf{Bij} \simeq \mathsf{FinSet}$. *In particular, one can extract a type* $\langle x \rangle : \mathsf{Type}$ *from each* $x : \mathsf{Bij}$.

A small type of finite bags is defined by replacing FinSet with Bij.

$$\mathsf{Bag}\,X =_{\mathsf{df}} \sum(x : \mathsf{Bij}).\ \langle x \rangle \to X$$

▶ **Proposition 19** (`BagToteEquiv`). *For any type* X*, the equivalence of Proposition 18 extends to an equivalence* $\mathsf{Bag}\,X \simeq \mathsf{Tote}\,X$ *natural in* X.

Combining the above with Theorem 17 yields the follows convenient characterization:

▶ **Corollary 20** (`TruncBagFMSetEquiv`). *For any* X, $\|\mathsf{Bag}\,X\|_2 \simeq \mathsf{FMSet}\,X$ *naturally in* X.

6 The Final Coalgebra in Groupoids

When defined this way, it is immediate that Bag is the polynomial functor associated to the *container* $(\mathsf{Bij}, \langle \_ \rangle)$ in the sense of [3, Definition 2]. Crucially, [3, Theorem 7] proves that for such functors, the ω-limit is the carrier of the final coalgebra; independently of the homotopy level of the container it is associated to. Therefore $\mathsf{L}_{\mathsf{Bag}}$ carries the structure of a final Bag-coalgebra, even though Bij is not a set:

▶ **Theorem 21** (isLimitPreservingBag). *The map* $\mathsf{pres}_{\mathsf{Bag}} : \mathsf{Bag}\ \mathsf{L}_{\mathsf{Bag}} \to \mathsf{L}^{\mathsf{sh}}_{\mathsf{Bag}}$ *is an equivalence of groupoids.*

▶ **Corollary 22** ([3, Theorem 7]). $\mathsf{pres}^{-1}_{\mathsf{Bag}} \circ \mathsf{shift} : \mathsf{L}_{\mathsf{Bag}} \to \mathsf{Bag}\ \mathsf{L}_{\mathsf{Bag}}$ *is the final* Bag*-coalgebra.*

We refer to the formalization of [3] for a proof of Corollary 22.

Iterated application of Corollary 20 and Theorem 2 shows that $\|\mathsf{Bag}^n\ 1\|_2$ is equivalent to $\mathsf{FMSet}^n\ 1$ for all n (IterTruncBagFMSetEquiv). One might wonder whether similarly the set-truncation of $\mathsf{L}_{\mathsf{Bag}}$ delivers the final coalgebra of FMSet. We are able to show that $\|\mathsf{L}_{\mathsf{Bag}}\|_2$ is a fixpoint of FMSet. But to prove finality, we require the additional assumption of the axiom of choice and a "higher" version $\mathsf{AC}_{3,2}$ of the axiom of choice [26, Exercise 7.8],[2] which states that for a set X and a groupoid-valued type family Y on X, the following type is inhabited: $((x : X) \to \|Y\ x\|_2) \to \|(x : X) \to Y\ x\|_2$.

▶ **Theorem 23** (FMSetFixpointTruncBagLim). *The set-truncation of* $\mathsf{L}_{\mathsf{Bag}}$ *is a fixpoint of* FMSet*,* i.e. *there is an equivalence* $\mathsf{FMSet}\ \|\mathsf{L}_{\mathsf{Bag}}\|_2 \simeq \|\mathsf{L}_{\mathsf{Bag}}\|_2$.

Proof. The equivalence is obtained from the composition

$$\mathsf{FMSet}\ \|\mathsf{L}_{\mathsf{Bag}}\|_2 \overset{\alpha}{\simeq} \mathsf{FMSet}\ \mathsf{L}_{\mathsf{Bag}} \overset{\beta}{\simeq} \|\mathsf{Bag}\ \mathsf{L}_{\mathsf{Bag}}\|_2 \overset{\gamma}{\simeq} \|\mathsf{L}_{\mathsf{Bag}}\|_2$$

where α is invariance of FMSet under set-truncation (Theorem 2), β follows from Corollary 20, and γ follows from Theorem 21. ◀

Let $\mathsf{coalg}_{\mathsf{FMSet}}$ be the coalgebra underlying the equivalence of Theorem 23.

▶ **Theorem 24** (FMSetFinalCoalgebra). *Assuming axiom of choice and* $\mathsf{AC}_{3,2}$*,* $\|\mathsf{L}_{\mathsf{Bag}}\|_2$ *is the final coalgebra of* FMSet *in the category of sets.*

Proof. Let $c : X \to \mathsf{FMSet}\ X$ be a coalgebra, which by Corollary 20 is equivalent to having a function $c' : X \to \|\mathsf{Bag}\ X\|_2$. Applying $\mathsf{AC}_{3,2}$ on c' gives $c'' : \|X \to \mathsf{Bag}\ X\|_2$. Invoking $\mathsf{rec}_{\|X \to \mathsf{Bag}\ X\|_2}$ on c'', we receive $g : X \to \mathsf{Bag}\ X$. From the finality of $\mathsf{L}_{\mathsf{Bag}}$ in Corollary 22, there exists a unique Bag-coalgebra morphism $f^* : X \to \mathsf{L}_{\mathsf{Bag}}$ and we define $f\ x =_{\mathsf{df}} |f^*\ x|_2$. The function f is the desired unique FMSet-coalgebra morphism between (X, c) and $(\|\mathsf{L}_{\mathsf{Bag}}\|_2, \mathsf{coalg}_{\mathsf{FMSet}})$. The proof of uniqueness uses an application of the axiom of choice. We refer to the formalization for details. ◀

In the absence of the axiom of choice and $\mathsf{AC}_{3,2}$, the type $\|\mathsf{L}_{\mathsf{Bag}}\|_2$ can still be used to give semantics to transition systems with *finite* set of states.

▶ **Proposition 25** (uniqueCoalgMorphismFinCarrier). *Given any* $n : \mathbb{N}$ *and a coalgebra* $c : \mathsf{Fin}\ n \to \mathsf{FMSet}(\mathsf{Fin}\ n)$*, there exists a unique coalgebra morphism from* $(\mathsf{Fin}\ n, c)$ *to* $(\|\mathsf{L}_{\mathsf{Bag}}\|_2, \mathsf{coalg}_{\mathsf{FMSet}})$.

This is true since $\mathsf{AC}_{3,2}$ holds when X is equivalent to $\mathsf{Fin}\ n$, it follows from the "finite choice" principle in Lemma 1. The particular instance of the axiom of choice used in the proof of Theorem 24 also holds when $X \simeq \mathsf{Fin}\ n$.

[2] We use 0-based indexing of h-levels, while [26] uses -2-based indexing, so our $\mathsf{AC}_{3,2}$ is their $\mathsf{AC}_{1,0}$.

7 Other Analytic Functors

The formulation FMSet of the finite bag functor exposes this as an analytic functor [15, 12], which differs from a polynomial functor in that the type of tuples $\mathsf{Fin}\ n \to X$ is quotiented by the relation induced by the action of the symmetric group on $\mathsf{Fin}\, n$. Other analytic functors arise by choosing a different subgroup of the symmetric group. For example, picking the subgroup of cyclic permutations delivers the functor of cyclic lists, while taking the trivial subgroup allows us to recover the list functor.

In type theory analytic functors can be seen as instances of the functors associated to the *quotient containers* of Abbott *et al.* [1]. A quotient container is a triple consisting of a type A, a family $B : A \to \mathsf{Type}$ and a propositionally-valued family $P : \forall\{a\}.\ B\, a \simeq B\, a \to \mathsf{Type}$ closed under identity, inverses and composition of equivalences. The associated functor is:

$$\mathsf{F}_{A,B,P}\ X =_{\mathsf{df}} \sum(a : A).\ (B\, a \to X)\ /_2\ \mathsf{Act}\ P\, a$$

where the relation $\mathsf{Act}\ P\, a$ is

$$\mathsf{Act}\ P\, a\ v\ w =_{\mathsf{df}} \exists(\sigma : B\, a \simeq B\, a).\ P\, \sigma \times (v = w \circ \sigma)$$

The type $\mathsf{F}_{A,B,P}\ X$ is a set whenever the type of shapes A is a set. The functor FMSet corresponds to the instance where $A =_{\mathsf{df}} \mathbb{N}$, $B =_{\mathsf{df}} \mathsf{Fin}$ and $P\, \sigma =_{\mathsf{df}} 1$.

We know that the construction of the final coalgebra as an ω-limit in the category of sets for a general analytic functor $\mathsf{F}_{A,B,P}$ is constructively problematic, since it is already problematic for FMSet. Nevertheless, one can ask if a result like Theorem 11 is valid for any $\mathsf{F}_{A,B,P}$. We do not know how to generally define a section for the function $\mathsf{pres}_{\mathsf{F}} : \mathsf{F}_{A,B,P}(\lim_n(\mathsf{F}^n_{A,B,P}\, 1)) \to \lim_n(\mathsf{F}^{1+n}_{A,B,P}\, 1)$. But we believe the surjectivity of $\mathsf{pres}_{\mathsf{F}}$ to be provable under the assumption of the axiom of countable choice. The proof of Theorem 11 relies on Propositions 4 and 5, which are very specific properties of the finite bag functor. The employment of these propositions can be seen as the invocation of two specific instances of the axiom of countable choice, which happen to hold in the case of FMSet.

Each quotient container (A, B, P) also specifies a polynomial functor $\mathsf{G}_{A,B,P}$ valued in groupoids, akin to the functor Bag. First, the small HIT construction of the groupoid of finite types Bij can be generalized:

$$\frac{a : A}{\mathsf{obj}\, a : \mathsf{U}_{A,B,P}} \qquad \frac{a : A \quad \alpha : Ba \simeq Ba \quad p : P\, \alpha}{\mathsf{hom}\ \alpha\, p : \mathsf{obj}\, a = \mathsf{obj}\, a}$$

$$\frac{a : \mathbb{N} \qquad \alpha, \beta : B\, a \simeq B\, a \qquad p : P\, \alpha \qquad q : P\, \beta}{\mathsf{hom}\, (\beta \circ \alpha)\, (\mathsf{Pcomp}\ p\, q) = \mathsf{hom}\ \alpha\, p \bullet \mathsf{hom}\ \beta\, q} \qquad \frac{}{\mathsf{isGroupoid}\ \mathsf{U}_{A,B,P}}$$

Above Pcomp is the closure of P with respect to composition of equivalences. It is possible to prove that hom also preserves identities and inverses. When B is valued in sets, there is a function $\langle - \rangle : \mathsf{U}_{A,B,P} \to \mathsf{Type}$ extracting a set, so that $\langle \mathsf{obj}\, a \rangle =_{\mathsf{df}} B\, a$. The functor $\mathsf{G}_{A,B,P}$ is

$$\mathsf{G}_{A,B,P}\, X =_{\mathsf{df}} \sum(x : \mathsf{U}_{A,B,P}).\ \langle x \rangle \to X$$

Since $\mathsf{G}_{A,B,P}$ is a polynomial functor, its final coalgebra can be constructed as an ω-limit, as in Theorem 21 and Corollary 22. We conjecture that the latter can be related to $\mathsf{F}_{A,B,P}$ similarly to how Bag and FMSet are related via Corollary 20: if B is injective, then $\|\mathsf{G}_{A,B,P}\, X\|_2 \simeq \mathsf{F}_{A,B,P}\, X$. Notice that Fin is an injective type family, which the proof of $\mathsf{Bij} \simeq \mathsf{FinSet}$ (Proposition 18) crucially depends on.

8 Conclusions

We looked at various definitions of the finite bag functor, valued in sets and in groupoids, and constructions of their final coalgebras. When working with set-based definitions, the set-theoretic constructions as ω-limit of the terminal chain in (3) and as quotient of the final List-coalgebra are not directly replicable in HoTT, since they imply the validity of classical principles like LLPO. We are at least able to salvage the weak preservation of the ω-limit. The situation is brighter when working with the groupoid-based definition. The latter is a polynomial functor, thus has the final coalgebra given by the ω-limit of the terminal chain.

Our conclusion is in line with the one of Kock [16] and Finster *et al.* [10]: the bag functor is better behaved when valued in groupoids instead of sets, especially from the perspective of final semantics. This seems to indicate that the denotational semantics of "resource-sensitive" computations is better performed using groupoids instead of sets (switching from categorical to bicategorical semantics). In particular, the syntax of process calculi such as CCS, or term calculi for linear logic, could be defined directly as a groupoid, *i.e.* structural congruences could be treated as data instead of property. We plan to properly investigate this connection to programming language semantics in future work, along the lines of [9]. For this endeavor, it will also be necessary to study the final coalgebra of combinations of the bag functor with other functors *e.g.* formalizing the presence of labels or actions in the transition relation.

Cubical Agda allows the definition of coinductive types with HITs appearing in the codomain of destructors. For example, it is possible to define the following coinductive record:

```
record cLim-FCM : Type where
  coinductive
  field
    unfold : FCM cLim-FCM
```

It is moreover possible to prove that this type is the final coalgebra of the set-based finite bag functor. The proof is similar to the one given for the finite powerset functor [29, Theorem 2]. Definitions such as `cLim-FCM` are an experimental feature of Cubical Agda, since the interaction of coinductive types and HITs has not yet been investigated (only for some M-types [30], which are definable internally in HoTT anyway). We believe that such definitions could be motivated by looking at recent work by Kristensen *et al.* [18], which seems to indicate that the final coalgebra of functors with action on objects given as a HIT, such as FCM, should be definable as the *strict* ω-limit of the chain in (3) in the cubical set model. Strictness means that the limit-property holds on the nose, not only up-to path equality.

References

1 Michael Abbott, Thorsten Altenkirch, Neil Ghani, and Conor McBride. Constructing polymorphic programs with quotient types. In Dexter Kozen and Carron Shankland, editors, *Proc. of 7th Int. Conf. on Mathematics of Program Construction, MPC'04*, volume 3125 of *LNCS*, pages 2–15. Springer, 2004. doi:10.1007/978-3-540-27764-4_2.

2 Jirí Adámek and Václav Koubek. On the greatest fixed point of a set functor. *Theoretical Computer Science*, 150(1):57–75, 1995. doi:10.1016/0304-3975(95)00011-K.

3 Benedikt Ahrens, Paolo Capriotti, and Régis Spadotti. Non-wellfounded trees in Homotopy Type Theory. In Thorsten Altenkirch, editor, *Proc. of 13th Int. Conf. on Typed Lambda Calculi and Applications, TLCA'15*, volume 38 of *LIPIcs*, pages 17–30. Schloss Dagstuhl, 2015. doi:10.4230/LIPICS.TLCA.2015.17.

4 Michael Barr. Terminal coalgebras in well-founded set theory. *Theoretical Computer Science*, 114(2):299–315, 1993. doi:10.1016/0304-3975(93)90076-6.

5 Gilles Barthe, Venanzio Capretta, and Olivier Pons. Setoids in type theory. *Journal of Functional Programming*, 13(2):261–293, 2003. doi:10.1017/S0956796802004501.

6 Douglas Bridges and Fred Richman. *Varieties of Constructive Mathematics*. Cambridge University Press, 1987. doi:10.1017/cbo9780511565663.

7 Paolo Capriotti, Nicolai Kraus, and Andrea Vezzosi. Functions out of higher truncations. In Stephan Kreutzer, editor, *Proc. of 24th EACSL Ann. Conf. on Computer Science Logic, CSL'15*, volume 41 of *Leibniz International Proceedings in Informatics*, pages 359–373. Schloss Dagstuhl, 2015. doi:10.4230/LIPICS.CSL.2015.359.

8 Vikraman Choudhury and Marcelo Fiore. Free commutative monoids in homotopy type theory. In *Proc. of 38th Conf. on Mathematical Foundations of Programming Semantics (MFPS XXXVIII)*, volume 1. Centre pour la Communication Scientifique Directe (CCSD), 2023. doi:10.46298/entics.10492.

9 Vikraman Choudhury, Jacek Karwowski, and Amr Sabry. Symmetries in reversible programming: from symmetric rig groupoids to reversible programming languages. *Proc. of the ACM on Programming Languages*, 6(POPL):1–32, 2022. doi:10.1145/3498667.

10 Eric Finster, Samuel Mimram, Maxime Lucas, and Thomas Seiller. A cartesian bicategory of polynomial functors in homotopy type theory. In Ana Sokolova, editor, *Proc. of 37th Conf. on Mathematical Foundations of Programming Semantics, MFPS'21*, volume 351 of *EPTCS*, pages 67–83, 2021. doi:10.4204/EPTCS.351.5.

11 Dan Frumin, Herman Geuvers, Léon Gondelman, and Niels van der Weide. Finite sets in homotopy type theory. In June Andronick and Amy P. Felty, editors, *Proc. of 7th ACM SIGPLAN Int. Conf. on Certified Programs and Proofs, CPP'18*, pages 201–214. ACM, 2018. doi:10.1145/3167085.

12 Ryu Hasegawa. Two applications of analytic functors. *Theoretical Computer Science*, 272(1-2):113–175, 2002. doi:10.1016/S0304-3975(00)00349-2.

13 Ichiro Hasuo, Kenta Cho, Toshiki Kataoka, and Bart Jacobs. Coinductive predicates and final sequences in a fibration. In Dexter Kozen and Michael W. Mislove, editors, *Proc. of the 29th Conf. on the Mathematical Foundations of Programming Semantics, MFPS'13*, volume 298 of *ENTCS*, pages 197–214. Elsevier, 2013. doi:10.1016/j.entcs.2013.09.014.

14 Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016. doi:10.1017/CBO9781316823187.

15 André Joyal. Foncteurs analytiques et espèces de structures. In *Combinatoire énumérative*, pages 126–159. Springer Berlin Heidelberg, 1986. doi:10.1007/bfb0072514.

16 Joachim Kock. Data Types with Symmetries and Polynomial Functors over Groupoids. In Ulrich Berger and Michael Mislove, editors, *Proc. of 28th Conf. on Mathematical Foundations of Programming Semantics, MFPS'12*, volume 286 of *ENTCS*, pages 351–365. Elsevier, 2012. doi:10.1016/j.entcs.2013.01.001.

17 Nicolai Kraus, Martín Escardó, Thierry Coquand, and Thorsten Altenkirch. Notions of Anonymous Existence in Martin-Löf Type Theory. *Logical Methods in Computer Science*, 13(1), 2017. doi:10.23638/LMCS-13(1:15)2017.

18 Magnus Baunsgaard Kristensen, Rasmus Ejlers Møgelberg, and Andrea Vezzosi. Greatest HITs: Higher inductive types in coinductive definitions via induction under clocks. In Christel Baier and Dana Fisman, editors, *Proc. of 37th Ann. ACM/IEEE Symp. on Logic in Computer Science, LICS'22*, pages 42:1–42:13. ACM, 2022. doi:10.1145/3531130.3533359.

19 Paul Blain Levy. Similarity quotients as final coalgebras. In Martin Hofmann, editor, *Proc. of 14th Int. Conf on Foundations of Software Science and Computational Structures, FoSSaCS'11*, volume 6604 of *LNCS*, pages 27–41. Springer, 2011. doi:10.1007/978-3-642-19805-2_3.

20 Nuo Li. *Quotient types in type theory*. PhD thesis, University of Nottingham, UK, 2015. URL: http://eprints.nottingham.ac.uk/28941/.

21 Mark Mandelkern. Constructively complete finite sets. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 34(2):97–103, 1988. doi:10.1002/malq.19880340202.

22 Stefano Piceghello. *Coherence for Monoidal and Symmetric Monoidal Groupoids in Homotopy Type Theory*. PhD thesis, University of Bergen, Norway, 2021. URL: https://hdl.handle.net/11250/2830640.

23 Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000. doi:10.1016/S0304-3975(00)00056-6.

24 Kristina Sojakova. *Higher Inductive Types as Homotopy-Initial Algebras*. PhD thesis, Carnegie Mellon University, USA, 2016. URL: http://reports-archive.adm.cs.cmu.edu/anon/anon/usr0/ftp/home/ftp/2016/CMU-CS-16-125.pdf.

25 The agda/cubical development team. The agda/cubical library, 2018. URL: https://github.com/agda/cubical/.

26 The Univalent Foundations Program. *Homotopy type theory: Univalent foundations of mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

27 Daniele Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *Proc. of 12th Ann. IEEE Symp. on Logic in Computer Science, LICS'97*, pages 280–291. IEEE Computer Society, 1997. doi:10.1109/LICS.1997.614955.

28 Niccolò Veltri and Niels van der Weide. Constructing higher inductive types as groupoid quotients. *Logical Methods in Computer Science*, 17(2), 2021. doi:10.23638/LMCS-17(2:8)2021.

29 Niccolò Veltri. Type-theoretic constructions of the final coalgebra of the finite powerset functor. In Naoki Kobayashi, editor, *Proc. of 6th Int. Conf. on Formal Structures for Computation and Deduction, FSCD'21*, volume 195 of *LIPIcs*, pages 22:1–22:18. Schloss Dagstuhl, 2021. doi:10.4230/LIPICS.FSCD.2021.22.

30 Andrea Vezzosi, Anders Mörtberg, and Andreas Abel. Cubical Agda: A dependently typed programming language with univalence and higher inductive types. *Proc. of the ACM on Programming Languages*, 3(ICFP):1–29, 2019. doi:10.1145/3341691.

31 Vladimir Voevodsky. An experimental library of formalized mathematics based on the univalent foundations. *Mathematical Structures in Computer Science*, 25(5):1278–1294, 2015. doi:10.1017/s0960129514000577.

32 James Worrell. On the final sequence of a finitary set functor. *Theoretical Computer Science*, 338(1-3):184–199, 2005. doi:10.1016/j.tcs.2004.12.009.

33 Brent Yorgey. *Combinatorial Species and Labelled Structures*. PhD thesis, University of Pennsylvania, 2014. URL: http://ozark.hendrix.edu/~yorgey/pub/thesis.pdf.

34 Brent A. Yorgey. Species and functors and types, oh my! In Jeremy Gibbons, editor, *Proc. of 3rd ACM Symp. on Haskell, Haskell'10*, pages 147–158. ACM, 2010. doi:10.1145/1863523.1863542.