W-TYPES IN SETOIDS

JACOPO EMMENEGGER

ABSTRACT. W-types and their categorical analogue, initial algebras for polynomial endofunctors, are an important tool in predicative systems to replace transfinite recursion on well-orderings. Current arguments to obtain W-types in quotient completions rely on assumptions, like Uniqueness of Identity Proofs, or on constructions that involve recursion into a universe, that limit their applicability to a specific setting. We present an argument, verified in Coq, that instead uses dependent W-types in the underlying type theory to construct W-types in the setoid model. The immediate advantage is to have a proof more type-theoretic in flavour, which directly uses recursion on the underlying W-type to prove initiality. Furthermore, taking place in intensional type theory and not requiring any recursion into a universe, it may be generalised to various categorical quotient completions, with the aim of finding a uniform construction of extensional W-types.

1. INTRODUCTION

The present paper is a contribution to the study of models of extensional properties in intensional type theories and is in particular concerned with W-types. The W-type constructor in Martin-Löf type theory [24, 25] produces an inductive type whose terms can be understood as well-founded trees. Alternatively, it may be viewed as the free term algebra for a, possibly infinitary, single-sorted signature. It may be used to construct in a uniform way several inductive types, like natural numbers or lists, it provides a predicative counterpart to the notion of well-ordering, and it may be used to give constructive justifications of certain theories of inductive definitions [29]. Furthermore, not only it is instrumental in constructing Aczel's model of Constructive Zermelo-Fraenkel set theory in Martin-Löf type theory [1], where sets are well-founded trees labelled by (small) types, but it also allows to interpret the Regular Extension Axiom which adds general inductive definitions to CZF [2].

Moerdijk and Palmgren identified a category-theoretic counterpart of W-types in initial algebras for polynomial endofunctors [26]. This notion has then been used in the context of predicative algebraic set theory to obtain models of constructive set theories and, more generally, for a model-theoretic analysis of predicative systems [27, 16, 5, 6, 7, 8]. More recently, Swan has shown how to simulate the small object argument using (a generalisation of) W-types in a locally cartesian closed pretopos, in order to construct algebraic weak factorisation systems [33].

Most of these constructions and applications of W-types, however, require some extensionality to hold in the type theory, which destroys the good computational properties of the intensional theory. For example, in Dybjer's proof that fixed points of certain type operators may be represented by W-types [13, 14], the identity reflection rule plays an essential role. Let us point out that such type operators are those where the type variable only appears on the right of function types. Hence, together with [2, 29], Dybjer's result inspired

the semantic characterisation of W-types and initial algebras for polynomial endofunctors by Moerdijk and Palmgren. It is of no surprise then that the current category-theoretic formulation of W-types takes place in a locally cartesian closed (l.c.c.) category, i.e. the category-theoretic counterpart of extensional type theory [32, 19].

Therefore, it is desirable to have a uniform construction of W-types for the great variety of quotient completions used for modelling extensional type theories into intensional ones, both from a type-theoretic standpoint, i.e. setoid models [20, 4, 21], and from a categorical one, namely exact completions [12, 10] and more general forms of quotient completions [22, 23, 9]. At present, the author is aware of constructions of W-types for two such quotient completions. The first one is due to van den Berg [5] and it applies to exact completions of categories with finite limits. But the setoid model in intensional Martin-Löf type theory is an instance of such completion if and only if UIP holds in the underlying type theory [15]. The second construction, instead, was formulated by Palmgren for the setoid model in intensional Martin-Löf type theory and then adapted by Bressan to Minimalist Type Theory [11]. It uses in an essential way recursion into a universe, hence it is not directly portable to a (predicative) categorical setting, like the internal language of a ΠW-pretopos, i.e. a locally cartesian closed pretopos with W-types.

This paper makes a first step towards this general goal, by providing a "small" construction of initial algebras for polynomial endofunctors in the e-category of small setoids Std. This construction is "small" in the sense that it does not require recursion into a universe and it is thus more suited for being reformulated in a categorical context. The initial observation is that, in order to deal with properties of an inductive type, it is useful to have a way to construct inductive predicates, i.e. families of inductive types. One possibility is to construct such predicates by recursion into a universe, like Palmgren did. Alternatively, we may consider the possibility of making the needed constructions *primitive* in the theory. It turns out that the predicates needed in the construction of initial algebras are instances of a generalised form of W-type constructor, the so-called *dependent W-type*, also known as tree type.

Dependent W-types were introduced by Petersson and Synek [31] to provide a constructor for general inductive data types, that is, families of mutually dependent inductive types (see also [28]). They generalise simple W-types in the sense that dependent ones give rise to free term algebras for multi-sorted signatures. Gambino and Hyland identified a category-theoretic counterpart in initial algebras for certain endofunctors, called *dependent polynomial endofunctor* [17]. They also proved that, in an l.c.c. category, dependent W-types can be obtained from non-dependent ones, thus providing justification for an analogous result in type theory claimed in [31]. An extensive study of (dependent) polynomial functors which includes the semantic of W-types, but goes well beyond that, is in [18].

When trying to construct initial algebras in a predicative and intensional setting, two steps are more demanding: identifying the object for the initial algebra, and constructing the morphism of algebras witnessing initiality. Defining the algebra map, as well as showing commutativity and uniqueness for the morphism of algebras are usually conceptually simpler. In the case of setoids, the first step amounts to constructing a partial equivalence relation on the underlying W-type W. In Palmgren's argument, this is done using recursion on the underlying W-type into a universe but, as we shall see, that relation is just an inductive family on the product $W \times W$.

For the construction of the morphism of algebras, a set-theoretic argument would consider a (well-founded) subtree relation and do transfinite recursion on it to obtain a function defined on all subtrees. An additional one step construction will then extend this to a function from all trees. Palmgren implemented this argument in type theory

by constructing the transitive closure of a relation using recursion into a universe, and applying it to obtain the setoid of subtrees. In this construction though, the analogy between the recursion principle of W-types and initiality of the algebra is hidden behind a thick layer of set-theoretic machinery.

Instead, we just consider the setoid of immediate subtrees $\mathsf{ImS}\, w$, for a tree $w : W$ in the underlying type of the (candidate) initial algebra s. These setoids are just the image factorisation of the branching functions of the tree w, and form what we call a *proof-irrelevant setoid family*, namely a family of setoids indexed by a setoid (W in this case), whose transport arrows between fibres do not depend on the proof of equality. Any algebra morphism from s to an algebra a with underlying setoid A determines by restriction a family of functions $k_w : \mathsf{ImS}\, w \Rightarrow A$ for $w : W$, satisfying a coherence condition, namely that the family $(k_w)_w$ is stable under transport of ImS. The converse is true as well: any coherent family determines a (unique) algebra morphism.

One drawback of this approach is that the immediate subtree relation is not transitive. In particular, contrary to what happens with the setoid of all subtrees, there is no algebra map on $\mathsf{ImS}\, w$. This means that, when trying to do recursion on w to obtain a function $\mathsf{ImS}\, w \Rightarrow A$, we may not say that such a function is an algebra morphisms. In turn, this blocks the recursive step, since we no longer know whether the family of functions obtained by the inductive hypothesis is coherent. In order to tackle this issue, we exploit dependent W-types once more. Let us call an extensional function $k : \mathsf{ImS}\, w \Rightarrow A$ *recursively defined over w* if it is obtained applying the above step to a coherent family $(k_s : \mathsf{ImS}\, s \Rightarrow A)_s$, for s immediate subtree of w, such that each of the functions k_s is also recursively defined. Since the functions k_s are over an immediate subtree, we may use dependent W-types to construct the type of recursively defined functions.

Using the elimination principle of dependent W-types, we prove in Lemma 3.11 that any two recursively defined function over the same w are equal. This uniqueness is the key fact to prove initiality. Firstly, it ensures that the property of being recursively defined is preserved by the above recursion step, so that we obtain for each $w : W$, a recursively defined function $k_w : \mathsf{ImS}\, w \Rightarrow A$. Secondly, it implies that the setoid of algebra morphisms from s to a is isomorphic to the setoid of families of recursively defined functions. With this characterisation, we may turn the family $(k_w)_w$ of recursively defined functions into an algebra morphism k. Uniqueness of recursively defined functions then yields uniqueness of k.

The next section contains a description of the type theory we shall be working with, and defines the basic concepts we shall be dealing with. Section 3 contains the proof that setoids in Martin-Löf type theory have initial algebras for polynomial endofunctors. It begins in Section 3.1 with the construction of the algebra s, Section 3.2 defines the family of immediate subtrees and describes the recursive step. The type of recursively defined functions in constructed in Section 3.3, where uniqueness of such functions is also proved. Section 3.4 contains the characterisation of algebra morphisms as families of recursively defined functions, and Section 3.5 concludes the proof of initiality. We conclude with some remarks on future work and possible extensions to the categorical setting.

2. Type-theoretic setting

Although the formalisation is done in Coq, we only consider a fragment similar to the logical framework of Martin-Löf type theory [28]. Hence we only assume existence of extensional $\sum$-types (which we call *records* as in Coq), $\prod$-types and a universe U, T, which is the universe Set in the Coq formalisation. In addition, we require the universe to be

FIGURE 1. Rules for W-types

$$\frac{A : \mathsf{U} \qquad B : A \to \mathsf{U}}{\mathsf{W}_{A,B} : \mathsf{U}} \text{ W-FORM} \qquad\qquad \frac{a : A \qquad f : B\,a \to \mathsf{W}_{A,B}}{\mathsf{sup}_{A,B}\,a\,f : \mathsf{W}_{A,B}} \text{ W-INTRO}$$

$$\frac{C : \mathsf{W}_{A,B} \to \mathsf{U} \qquad c : \prod_{(a:A)} \prod_{(f:B\,a \to \mathsf{W}_{A,B})} \left(\prod_{b:B\,a} C\,(f\,b) \right) \to C\,(\mathsf{sup}_{A,B}\,a\,f)}{\mathsf{rec}^{\mathsf{W}}_{A,B,C,c} : \prod_{w:\mathsf{W}_{A,B}} C\,w} \text{ W-ELIM}$$

$$\frac{a : A \qquad f : B\,a \to \mathsf{W}_{A,B}}{\mathsf{rec}^{\mathsf{W}}_{A,B,C,c}\,(\mathsf{sup}_{A,B}\,a\,f) \equiv c\,a\,f\,(\lambda b.\mathsf{rec}^{\mathsf{W}}_{A,B,C,c}\,(f\,b))} \text{ W-CONV}$$

closed under $\prod$-types, and to contain intensional $\sum$-types, intensional identity types $=_X$, the unit type $\mathbf{1}$ and dependent W-types, whose rules are described in the next section.

We interpret the logic according to propositions-as-types. However, we believe that our argument may be adapted to work also for other interpretations of logic, namely either through a type of propositions as in the Calculus of Constructions or in the Minimalist Type Theory, or using h-propositions as in Homotopy Type Theory.

Application of function terms shall be denoted by juxtaposition, $\equiv$ and $:=$ shall denote judgemental equalities and definitions, respectively. In order to make judgements more readable, we shall drop occurrences of the decoding function T and, for $A : \mathsf{U}$, write $a : A$ instead of $a : \mathsf{T}\,A$.

2.1. **Dependent W-types.** W-types provide a uniform way to construct inductively defined types. Similarly, dependent W-types allow to construct families of inductive types. Rules for W-types and dependent W-types in our type theory are in Figures 1 and 2, respectively. We shall often drop (some) subscripts if they are clear from the context.

Useful ways to regard W-types are through the set-theoretic interpretation of type theory: in such a context W-types represent well-founded labelled trees or, equivalently, free term algebras for infinitary single-sorted signatures. In the first case the set A is the set of names for the nodes, for each $a \in A$, the set $B\,a$ consists of the branches out of the node with name a. Trees are formed by providing a node $a \in A$ and attaching other trees to the branches in $B\,a$. This procedure is formally specified by functions $f \colon B\,a \to \mathsf{W}_{A,B}$, which provide the instruction to attach the tree $f(b)$ to the branch $b \in B\,a$. In the second case, the set A contains the function symbols of the signature, while (the cardinality of) $B\,a$ is the arity of symbol a. Terms are build out of function symbols according to composition instructions specified by functions $f \colon B\,a \to \mathsf{W}_{A,B}$.

Similarly, dependent W-types may be seen as free term algebras for infinitary multi-sorted signatures: sorts are terms of type I, terms in $A\,i$ are function symbols with codomain sort i while the function d maps each tuple i, a, b into the sort of the b-th argument of the function symbol a.

FIGURE 2. Rules for dependent W-types

$$\frac{I : \mathsf{U} \qquad A : I \to \mathsf{U}}{B : \prod_{i:I} A\,i \to \mathsf{U} \qquad d : \prod_{(i:I)} \prod_{(a:A\,i)} B\,i\,a \to I}{\mathsf{DW}_{I,A,B,d} : I \to \mathsf{U}} \; \text{DW-FORM}$$

$$\frac{i : I \qquad a : A\,i \qquad f : \prod_{b:B\,i\,a} \mathsf{DW}_{I,A,B,d}\,(d\,i\,a\,b)}{\mathsf{dsup}_{I,A,B,d}\,i\,a\,f : \mathsf{DW}_{I,A,B,d}\,i} \; \text{DW-INTRO}$$

$$\frac{C : \prod_{i:I} \mathsf{DW}_{I,A,B,d}\,i \to \mathsf{U}}{c : \prod_{(i:I)} \prod_{(a:A\,i)} \prod_{(f:\prod_{b:B\,i\,a} \mathsf{DW}_{I,A,B,d}\,(d\,i\,a\,b))} \left(\prod_{b:B\,i\,a} C\,(d\,i\,a\,b)\,(f\,b) \right) \to C\,i\,(\mathsf{dsup}_{I,A,B,d}\,i\,a\,f)}{\mathsf{rec}^{\mathsf{DW}}_{I,A,B,d,C,c} : \prod_{i:I} \prod_{w:\mathsf{DW}_{I,A,B,d}\,i} C\,i\,w} \; \text{DW-ELIM}$$

$$\frac{i : I \qquad a : A\,i \qquad f : \prod_{b:B\,i\,a} \mathsf{DW}_{I,A,B,d}\,(d\,i\,a\,b)}{\mathsf{rec}^{\mathsf{DW}}_{I,A,B,d,C,c}\,i\,(\mathsf{dsup}_{I,A,B,d}\,i\,a\,f) \equiv c\,i\,a\,f\,(\lambda b.\mathsf{rec}^{\mathsf{DW}}_{C,c}\,(d\,i\,a\,b)\,(f\,b))} \; \text{DW-CONV}$$

Notice that, in order for this procedure to produce a non-empty set of trees (resp. terms), there must be at least one node with no branches (resp. at least a constant). This is reflected in type theory: if we take $B : A \to \mathsf{U}$ to be the constant family with value the unit type, then we will not be able to produce a closed term of type $\mathsf{W}_{A,B}$. In fact it is not difficult to see that, whenever B is a non-empty family, $\mathsf{W}_{A,B}$ is type-theoretically equivalent to the empty type $\mathsf{0}$. The meaning of non-empty can be understood either as $\prod_{a:A}(B\,a \to \mathsf{0}) \to \mathsf{0}$ in plain Martin-Löf type theory, or as the variant using propositional truncation $\prod_{a:A} ||B\,a||$, if available.

Using elimination, we may define function terms $\mathsf{n} : \mathsf{W}_{A,B} \to A$ and $\mathsf{b} : \prod_{w:\mathsf{W}_{A,B}} B\,(\mathsf{n}\,w) \to \mathsf{W}_{A,B}$ such that $\mathsf{n}\,(\mathsf{sup}\,a\,f) \equiv a$ and $\mathsf{b}\,(\mathsf{sup}\,a\,f) \equiv f$. Similarly, for every $i : I$ we have

$$\mathsf{dn}_i : \mathsf{DW}\,i \to A\,i \qquad \mathsf{db}_i : \prod_{(w:\mathsf{DW}\,i)} \prod_{(b:B\,i\,(\mathsf{dn}_i\,w))} \mathsf{DW}\,(s\,i\,(\mathsf{dn}_i\,w)\,b),$$

such that $\mathsf{dn}_i\,(\mathsf{dsup}\,i\,a\,f) \equiv a$ and $\mathsf{db}_i\,(\mathsf{dsup}\,i\,a\,f) \equiv f$.

2.2. **The e-category of setoids.** The notion of setoid allows the intensional type theory to represent extensional concepts. In fact, setoids provide a model of extensional type theory within the intensional one [20]. Recall that a setoid is informally defined as a type together with a type-theoretic equivalence relation on it. This can be made precise in

various different ways, e.g. considering partial relations, or Prop-valued relations if a type of proposition is available, or instead h-sets and mere relations in the sense of Homotopy Type Theory. In our context we define a setoid as follows.

Definition 2.1. A *setoid* X is a tuple $(X_0, \approx_X, r_X, s_X, t_X)$ where $X_0 : \mathsf{U}$, $\approx_X : X_0 \to X_0 \to \mathsf{U}$ and

$$r_X : \prod_{x:X_0} x \approx_X x, \qquad s_X : \prod_{x,x':X_0} x \approx_X x' \to x' \approx_X x,$$

$$t_X : \prod_{x,x',x'':X_0} x \approx_X x' \to x' \approx_X x'' \to x \approx_X x''.$$

The type of setoids is defined as a record on the types of $X_0, \approx_X, r_X, s_X$ and t_X and it is denoted Std.

Any small type $A : \mathsf{U}$ gives rise to two setoids: the discrete one, with equality $\lambda a, a'.a =_A a'$, and the codiscrete one, with equality $\lambda a, a'.\mathbf{1}$. Of course, the codiscrete setoid over any inhabited type is isomorphic to the discrete setoid on $\mathbf{1}$ in the category of setoids defined below. When we want to regard a type as a setoid, we shall use the discrete equality unless otherwise stated.

Functions between setoids are what one would expect.

Definition 2.2. Let X and Y be setoids. A function term $f_0 : X_0 \to Y_0$ is *extensional* (with respect to the equalities of X and Y) if there is a term of type

$$\mathrm{ext}(f) := \prod_{x,x':X_0} x \approx_X x' \to f_0(x) \approx_Y f(x).$$

The setoid $X \Rightarrow Y$ of *extensional functions* from X to Y has the type of extensional functions $\sum_{f_0:X_0 \to Y_0} \mathrm{ext}(f_0)$ as underlying type, and as equality the equivalence relation

$$f \approx_{X \Rightarrow Y} g := \prod_{x:X_0} f_0(x) \approx_Y g_0(x).$$

In fact, in the Coq implementation we found more convenient to define the type of extensional functions as a record rather than as a $\sum$-type. However this makes no essential difference.

In the rest of the paper we shall write $x : X$, for a setoid X, to mean $x : X_0$, and we shall often not distinguish between an extensional function $f : X \Rightarrow Y$ and its underlying function term $f_0 : X_0 \to Y_0$. We shall denote by $f \upharpoonright \alpha : f\,x \approx_Y f\,x'$ the proof of extensionality of f applied to $\alpha : x \approx_X x'$. Occasionally, we shall also find it convenient to drop the subscript from the equality of a setoid. We do not expect these abuses of notation to lead to confusion.

According to the idea of representing extensional concepts in intensional Martin-Löf type theory using setoids, it is natural to define a (locally small) category $\mathbb{A}$ to be given by a type of objects $\mathrm{Obj}_\mathbb{A}$ and setoids of arrows $\mathrm{Hom}_\mathbb{A} : \mathrm{Obj}_\mathbb{A} \to \mathrm{Obj}_\mathbb{A} \to \mathsf{Std}$, together with explicit function terms for identity and composition, where the latter has type

$$\prod_{a,b,c:\mathrm{Obj}_\mathbb{A}} \mathrm{Hom}_\mathbb{A}(b,c) \Rightarrow \mathrm{Hom}_\mathbb{A}(a,b) \Rightarrow \mathrm{Hom}_\mathbb{A}(a,c),$$

and with identity and associativity axioms formulated using equalities of setoids. A functor between two categories $\mathbb{A}$ and $\mathbb{B}$ consists of a function term F between their type of objects together with a term of type

$$\prod_{a,a':\mathrm{Obj}_\mathbb{A}} \mathrm{Hom}_\mathbb{A}(a,a') \Rightarrow \mathrm{Hom}_\mathbb{B}(F\,a, F\,a').$$

We shall denote the action of a functor F on an arrow $\alpha : \mathsf{Hom}_{\mathbb{A}}(a, a')$ as $F_\alpha : \mathsf{Hom}_{\mathbb{B}}(F\,a, F\,a')$. One may define a notion of natural transformation between functors and form the category $\mathsf{Fun}(\mathbb{A}, \mathbb{B})$ of functors between $\mathbb{A}$ and $\mathbb{B}$.

These formulations of the notions of category and functor in type theory are sometimes referred to as *e-category* and *e-functor*, in order to distinguish them from other possible formulations. However, since these are the only formulations that we shall consider in intensional type theory, we shall just say category and functor to mean e-category and e-functor.

Within the type theory we may construct a category of setoids [30], also denoted Std, whose type of objects is Std and whose setoid family of arrows is given by $\lambda X, Y.X \Rightarrow Y$. Identity and composition are defined in the obvious way, the latter shall be denoted as $g \circ f$. Since the universe of the underlying type theory contains 1 and is closed under $\sum$ and $\prod$-types, this category is locally cartesian closed [19, 15].

In Std, we may define for $f : B \Rightarrow A$ and $a : A$ the *setoid fibre* whose underlying type is $\sum_{b:B} f(b) \approx_A a$ and whose equality is $(b, \alpha) \approx (b', \alpha') := b \approx_B b'$. This assignment gives rise to a setoid family over A, i.e. a functor $f^\# : \mathsf{Fun}(A^\#, \mathsf{Std})$, where $A^\#$ is the discrete category on the setoid A: its type of objects is A_0 and, for $a, a' : A$, its setoid of arrows from a to a' is the codiscrete setoid on the type $a \approx_A a'$. Since $A^\#$ is a groupoid, each term $\alpha : a \approx_A a'$ gives rise to an isomorphism $f^\#_\alpha : f^\# a \Rightarrow f^\# a'$. Furthermore, $A^\#$ is posetal (but not skeletal), hence such isomorphism does not depend on equality terms, i.e. $f^\#_\alpha \approx f^\#_{\alpha'}$ for $\alpha, \alpha' : a \approx_A a'$, and $f^\#_\alpha \approx id_{f^\# a}$ for any $\alpha : a \approx_A a$.

The converse is also true: any functor $B : \mathsf{Fun}(A^\#, \mathsf{Std})$ gives rise to an extensional function $(\sum_{a:A} B\,a, \approx) \Rightarrow A$ whose underlying function term is the first projection and where equality on the domain is

$$(a, b) \approx (a', b') := \sum_{\alpha:a\approx_A a'} B_\alpha\, b \approx_{B\,a'} b'.$$

For such a functor B, we shall refer to its the action on α as transport along α. We shall also abbreviate $B_\alpha\, b \approx_{B\,a'} b'$ as $b \approx_\alpha b'$.

It is of no surprise that this correspondence between extensional functions and setoid families gives rise to an equivalence of categories between the slice Std/A and the category of setoid families $\mathsf{Fam}A := \mathsf{Fun}(A^\#, \mathsf{Std})$.

2.3. **Polynomial functors and W-types.** The category-theoretic analogue of W-types are initial algebras for polynomial endofunctors [26]. Let $f : B \to A$ be an arrow in a locally cartesian closed category $\mathbb{C}$, then we may define a functor $P_f : \mathbb{C} \to \mathbb{C}$, the *polynomial endofunctor* associated to f, which maps an object X into $\sum_A \prod_f (X \times B)$. An *algebra* for P_f is given by an object X and an arrow $s_X : P_f X \to X$, called *algebra map.* and such an algebra is *initial* if for any other algebra $t_Y : P_f Y \to Y$, there is a unique $h : X \to Y$ such that $t_X \circ (P_f\, h) = h \circ s_X$. It is a well-known result by Lambek that the algebra map of an initial algebra is invertible.

In extensional type theory with one universe, the (internal) category of small types and function terms is locally cartesian closed if the universe has 1, $\sum$ and $\prod$ types. Hence we may consider polynomial endofuctors for each function term $f : B \to A$. An initial algebra is then given by the W-type of the family $f^{-1} := \lambda a. \sum_{b:B} f(b) =_A a$ with algebra map

$$(a, k) : \sum_{a:A} \left(f^{-1}(a) \to \mathsf{W}_{A,f^{-1}} \right) \longmapsto \mathsf{sup}(a, k) : \mathsf{W}_{A,f^{-1}}.$$

It is not difficult to see that initiality of $(\mathsf{W}_{A,f^{-1}}, \mathsf{sup})$ amounts exactly to the recursion principle of $\mathsf{W}_{A,f^{-1}}$ [26].

In Std, instead of defining polynomial functors associated to extensional functions, we prefer to work with polynomial functors defined from setoid families: in light of the equivalence between extensional functions and setoid families, this makes no difference.

Definition 2.3. Let $B : \mathsf{Fam}\,A$ be a setoid family. The *polynomial functor associated to B* is defined on $X : \mathsf{Std}$ as

$$\mathsf{P}_B\,X := \Big(\sum_{a:A}(B\,a \Rightarrow X),\, \approx_{\mathsf{P}_B X} \Big), \qquad (a,k) \approx_{\mathsf{P}_B X} (a',k') := \sum_{\alpha:a\approx a'} k \approx k' \circ B_\alpha$$

$$(\mathsf{P}_B\,f)(a,k) := (a, f \circ k), \qquad \text{for } f : X \Rightarrow Y.$$

We shall say that an endofunctor on Std is polynomial if it is naturally isomorphic to a polynomial functor associated to a setoid family B.

3. Initial algebras in setoids

3.1. **The algebra of extensional trees.** We proceed now to construct a setoid W and a P_B-algebra structure on it. Let W_{A_0,B_0} be the W-type constructed on $A_0 : \mathsf{U}$ and the type family $B_0 : A_0 \to \mathsf{U}$, for $A : \mathsf{Std}$ and $B : \mathsf{Fam}\,A$. A term in $\mathsf{P}_B\,\mathsf{W}_{A_0,B_0}$ consists of a pair (a,k) where $a : A_0$ and $k : B_0\,a \to \mathsf{W}_{A_0,B_0}$. Hence sup gives rise to a function term $\mathsf{P}_B\,\mathsf{W}_{A_0,B_0} \to \mathsf{W}_{A_0,B_0}$. We then have to construct a partial equivalence relation $\approx_W$ on W_{A_0,B_0} such that this term will eventually be an isomorphism of setoids, that is, a relation such that

$$\mathsf{sup}\,a\,f \approx_W \mathsf{sup}\,a'\,f' \longleftrightarrow \sum_{\alpha:a\approx_A a'} f \approx f' \circ B_\alpha$$

$$\longleftrightarrow \sum_{\alpha:a\approx_A a'} \prod_{\substack{b,b' \\ b\approx_\alpha b'}} f\,b \approx_W f'\,b'.$$

It is tempting to take this condition itself as the definition of $\approx_W$, since on the right-hand side $\approx_W$ occurs only on immediate subtrees. Indeed, this relation can be defined by recursion on the inductive type W_{A_0,B_0} into the universe U. There is however a more elementary construction, that does not require any elimination into a universe, and involve dependent W-types.

Define a relation on W_{A_0,B_0} as a dependent W-type with indices in $\mathsf{W}_{A_0,B_0} \times \mathsf{W}_{A_0,B_0}$ as follows: let

$$I := \mathsf{W}_{A_0,B_0} \times \mathsf{W}_{A_0,B_0}, \qquad X\,(w,w') := \mathsf{n}\,w \approx_A \mathsf{n}\,w', \qquad Y\,(w,w')\,\alpha := \sum_{b,b'} b \approx_\alpha b',$$

$$d\,(w,w')\,\alpha\,(b,b',\beta) := (\mathsf{b}\,w\,b, \mathsf{b}\,w'\,b') : I,$$

the relation then is

$$\mathsf{W}_B^{\mathsf{per}}\,w\,w' := \mathsf{DW}_{I,X,Y,d}\,(w,w').$$

Lemma 3.1. *Let $w, w' : \mathsf{W}_{A_0,B_0}$. Then $\mathsf{W}_B^{\mathsf{per}}\,w\,w'$ is inhabited if and only if there are*

$$\alpha : \mathsf{n}\,w \approx_A \mathsf{n}\,w' \qquad \text{and} \qquad \phi : \prod\Big(z : \sum_{b,b'} b \approx_\alpha b'\Big)\,\mathsf{W}_B^{\mathsf{per}}\,(\mathsf{b}\,w\,(\mathsf{pr}_1\,z))\,(\mathsf{b}\,w'\,(\mathsf{pr}_2\,z)).$$

Proof. If $\mathsf{W}_B^{\mathsf{per}}\,w\,w'$ is inhabited then α and ϕ are obtained from the (dependent) name and branches functions dn and db. Conversely, when $w \equiv \mathsf{sup}\,a\,k$ and $w' \equiv \mathsf{sup}\,a'\,k'$, a term of $\mathsf{W}_B^{\mathsf{per}}\,w\,w'$ is given by $\mathsf{dsup}\,(w,w')\,\alpha\,\phi$.   □

Proposition 3.2. *The type family* $\mathsf{W}_B^{\mathsf{per}} : \mathsf{W}_{A_0,B_0} \to \mathsf{W}_{A_0,B_0} \to \mathsf{U}$ *is a partial equivalence relation, that is, the following types are inhabited:*

$$\prod_{w,w':\mathsf{W}_{A_0,B_0}} \mathsf{W}_B^{\mathsf{per}} \, w \, w' \to \mathsf{W}_B^{\mathsf{per}} \, w' \, w,$$

$$\prod_{w,w',w'':\mathsf{W}_{A_0,B_0}} \mathsf{W}_B^{\mathsf{per}} \, w \, w' \to \mathsf{W}_B^{\mathsf{per}} \, w' \, w'' \to \mathsf{W}_B^{\mathsf{per}} \, w \, w''.$$

Proof. These terms are obtained from straightforward applications of the elimination rule for dependent W-types. Alternatively, one may use the previous Lemma and recursion on W_{A_0,B_0}. $\square$

Notice that $\mathsf{W}_B^{\mathsf{per}} \, w \, w$ is inhabited if and only if the branching function $\mathsf{b} \, w : B_0 \, (\mathsf{n} \, w) \to \mathsf{W}_{A_0,B_0}$ is extensional in the sense that

$$b \approx_{B \, (\mathsf{n} \, w)} b' \to \mathsf{W}_B^{\mathsf{per}} \, (\mathsf{b} \, w \, b) \, (\mathsf{b} \, w \, b')$$

is inhabited. In particular, $\mathsf{W}_B^{\mathsf{per}}$ is not in general reflexive. We shall say that a tree $w : \mathsf{W}_{A_0,B_0}$ is *extensional* if there is a term in $\mathsf{W}_B^{\mathsf{per}} \, w \, w$. We may form the setoid of extensional trees $W := (W_0, \approx_W) : \mathsf{Std}$, where

$$W_0 := \sum_{w:\mathsf{W}_{A_0,B_0}} \mathsf{W}_B^{\mathsf{per}} \, w \, w \qquad \text{and} \qquad (w, \_) \approx_W (w', \_) := \mathsf{W}_B^{\mathsf{per}} \, w \, w'.$$

We shall often leave the proof of reflexivity in $\mathsf{W}_B^{\mathsf{per}} \, w \, w$ implicit, and write $w : W$ to mean $w : \mathsf{W}_{A_0,B_0}$ and w extensional.

The name and branches term functions $\mathsf{n} : \mathsf{W}_{A_0,B_0} \to A$ and $\mathsf{b} : \prod_w B_0 \, (\mathsf{n} \, w) \to \mathsf{W}_{A_0,B_0}$ give rise to an extensional function $\mathsf{n} : W \Rightarrow A$ and a family of extensional functions $\mathsf{b} : \prod_{w:W} B \, (\mathsf{n} \, w) \Rightarrow W$ which we shall denote with the same symbols. This latter family is in fact itself extensional, since there is a term

$$\mathsf{extb} : \prod_{w,w'::W} \prod_{\gamma:w\approx_W w'} \mathsf{b} \, w \approx (\mathsf{b} \, w') \circ (B_{\mathsf{n}\vec{\mathsf{r}}\gamma}).$$

In particular, immediate subtrees of extensional trees are themselves extensional.

Lemma 3.3. *Let $w, w' : W$. Then $w \approx_W w'$ if and only if there are*

$$\alpha : \mathsf{n} \, w \approx_A \mathsf{n} w' \qquad \text{and} \qquad \phi : \prod_{b:B \, (\mathsf{n} \, w)} \mathsf{b} \, w \, b \approx_W (\mathsf{b} \, w' \, (B_\alpha \, b)).$$

Proof. One direction is proven applying $\mathsf{n} \, \vec{\mathsf{r}}$ and extb. Conversely, suppose $w \equiv \mathsf{sup} \, (a, f)$ and $w' \equiv \mathsf{sup} \, (a', f')$ and let $\alpha : a \approx a'$ and $\phi : \prod_b \mathsf{W}_B^{\mathsf{per}} \, (f_0 \, b) \, (f_0' \, (B_\alpha \, b))$. Hence a proof of $w \approx_W w'$ is given by $\mathsf{dsup}\_ \alpha \, \psi$, where

$$\psi : \prod \left(z : \sum_{b,b'} b \approx_\alpha b' \right) \mathsf{W}_B^{\mathsf{per}} \, (f_0 \, (\mathrm{pr}_1 \, z)) \, (f_0' \, (\mathrm{pr}_2 \, z))$$

is defined on (b, b', β) to be $\phi \, b : f_0 \, b \approx f_0' \, (B_\alpha \, b)$ concatenated with $f' \, \vec{\mathsf{r}} \, \beta : f_0' \, (B_\alpha \, b) \approx f_0' \, b'$. $\square$

We now show that the function term $\mathsf{sup} : \prod_{a:A} (B \, a \to \mathsf{W}_{A_0,B_0}) \to \mathsf{W}_{A_0,B_0}$ gives rise to an extensional function $\mathsf{s} : \mathsf{P}_B \, W \Rightarrow W$.

Lemma 3.4. *Let $a : A$, $f : B \, a \Rightarrow W$. Then there is $\mathsf{s} \, (a, f) : W$.*

Proof. Immediate from the observation that $\mathsf{sup}\,a\,f$ is extensional if and only if $f : B_0 \to W_0$ is extensional. We provide the required terms and leave the verification to the reader. Let $f_0 := \mathrm{pr}_1 f : B_0\,a \to \mathsf{W}_{A_0,B_0}$. Then $w := \mathsf{sup}\,a\,f_0 : \mathsf{W}_{A_0,B_0}$ and

$$\mathsf{dsup}\,(w,w)\,(\rho\,a)\,\varepsilon'_f : \mathsf{W}_B^{\mathsf{per}}\,w\,w,$$

where $\rho\,a$ is reflexivity on a and $\varepsilon'_f : \prod\left(z : \sum_{b,b'} b \approx_{B\,a} b'\right) \mathsf{W}_B^{\mathsf{per}}\,(f_0\,(\mathrm{pr}_1\,z))\,(f_0\,(\mathrm{pr}_2\,z))$ is obtained from extensionality of f. Hence

$$\mathsf{s}\,(a,f) := (\mathsf{sup}\,a\,f_0, \mathsf{dsup}\,\_\,(\rho\,a)\,\varepsilon'_f) : W. \qquad\square$$

Lemma 3.5. *The function term* $\mathsf{s} : \mathsf{P}_B\,W \to W$ *is extensional.*

Proof. To have two equal elements in the domain is to have $a, a' : A$, $f : B\,a \Rightarrow W$, $f' : B\,a' \Rightarrow W$, $\alpha : a \approx a'$ and $\phi : f \approx f' \circ B_\alpha$. Lemma 3.3 yields the claim. $\qquad\square$

We may now prove a necessary condition for the algebra map $\mathsf{s} : \mathsf{P}_B\,W \Rightarrow W$ to be initial, namely that it is invertible.

Proposition 3.6. *There is* $\mathsf{us} : W \Rightarrow \mathsf{P}_B\,W$ *such that* $\mathsf{s} \circ \mathsf{us} \approx id_W$ *and* $\mathsf{us} \circ \mathsf{s} \approx id_{\mathsf{P}_B\,W}$.

Proof. The function us maps a tree $w : W$ into its name and branching function:

$$\mathsf{us}\,w := (\mathsf{n}\,w, \mathsf{b}\,w) : \sum_{a:A}(B\,a \Rightarrow W).$$

The terms $\mathsf{n}\,\mathsf{\upharpoonright}$ and extb ensure its extensionality. The two equations follow unfolding the definitions of s and us. $\qquad\square$

3.2. **The family of immediate subtrees.** Let $C : \mathsf{Std}$ and $a_C : \mathsf{P}_B\,C \Rightarrow C$ be a P_B-algebra. Our aim is now to set up the machinery needed to construct the universal arrow $W \Rightarrow C$. This shall involve formulating the recursive step and isolating properties that ensure its applicability recursively.

Definition 3.7. Let $w : W$. The setoid of *immediate subtrees of w*, denoted $\mathsf{ImS}\,u$, has $B_0\,(\mathsf{n}w)$ as underlying type, and

$$b \approx_{\mathsf{ImS}\,u} b' := \mathsf{b}\,w\,b \approx_W \mathsf{b}\,w\,b'$$

as equality. The assignment $\mathsf{ImS}_\gamma\,s := B_{\mathsf{n}\mathsf{\upharpoonright}\gamma}\,s$, for $s : \mathsf{ImS}\,w$ and $\gamma : w \approx w'$, defines transport maps for ImS. Hence we obtain a setoid family $\mathsf{ImS} : \mathsf{Fam}W$, the *family of immediate subtrees*.

Those familiar with the exact completion construction will see that, for each $w : W$, the image factorisation of the branching function $\mathsf{b}\,w$ takes the form

$$\begin{array}{ccc}
B\,(\mathsf{n}\,w) & \xrightarrow{\ \ \mathsf{b}\,w\ \ } & W \\
& & \\
e_w := (id, (\mathsf{b}\,w)\mathsf{\upharpoonright}) \searrow & & \nearrow m_w := ((\mathsf{b}\,w)_0, id) \\
& \mathsf{ImS}\,w. &
\end{array}$$

where e_w and m_w denote the epi and mono arising from the factorisation of $\mathsf{b}\,w$. The underlying function term of e_w is the identity, while its proof of extensionality is just extensionality of $\mathsf{b}\,w$. The function m_w instead is the function term of $\mathsf{b}\,w$ together with the identity function as proof of extensionality.

The recursive step shall consist of constructing an extensional function $\mathsf{ImS}\,w \Rightarrow C$ out of a family of extensional functions $\mathsf{ImS}\,(\mathsf{b}\,w\,s) \Rightarrow C$, for $s : \mathsf{ImS}\,w$.

Definition 3.8. Let $w : W$. A family of extensional functions

$$F : \prod_{s : \mathsf{ImS}\, w} \mathsf{ImS}\,(\mathsf{b}\, w\, s) \Rightarrow C$$

is *coherent* if, for all $s, s' : \mathsf{ImS}\, w$ and $\sigma : s \approx_{\mathsf{ImS}\, w} s'$, $F\, s \approx (F\, s') \circ \mathsf{ImS}_\sigma$. We shall say that two coherent families F and F' are equal if $\prod_{s : \mathsf{ImS}\, w} F s \approx F's$ and denote with $\mathsf{CohMaps}\, w$ the setoid of coherent families of extensional functions.

In fact $\mathsf{CohMaps}$ is another setoid family over W, whose transport function for $\gamma : w \approx_W w'$

$$\mathsf{CohMaps}_\gamma : \mathsf{CohMaps}\, w \to \mathsf{CohMaps}\, w'$$

is defined on F and $s : \mathsf{ImS}\, w'$ as $(F\,(\mathsf{ImS}_{\gamma^{-1}}\, s)) \circ \mathsf{ImS}_{(\mathsf{extb}\, \gamma^{-1})}$. Hence to have $F \approx_\gamma F'$ is to have, for all $s : \mathsf{ImS}\, w$,

$$F\, s \approx_{(\mathsf{ImS}\,(\mathsf{n}\,(\mathsf{b}\, w\, s)) \Rightarrow C)} (F'\,(\mathsf{ImS}_\gamma\, s)) \circ \mathsf{ImS}_{(\mathsf{extb}\, \gamma)}.$$

Lemma 3.9 (Recursive step). *For every $w : W$ and $F : \mathsf{CohMaps}\, w$, there is*

$$\mathsf{recst}\, w\, F : \mathsf{ImS}\, w \Rightarrow C,$$

such that, for every $\gamma : w \approx_W w'$, $F : \mathsf{CohMaps}\, w$ and $F' : \mathsf{CohMaps}\, w'$ there is

$$\mathsf{extrecst}\, \gamma : F \approx_\gamma F' \longrightarrow \mathsf{recst}\, w\, F \approx (\mathsf{recst}\, w'\, F') \circ \mathsf{ImS}_\gamma.$$

Proof. The underlying function term of $\mathsf{recst}\, w\, F$ is defined on $s : \mathsf{ImS}\, w$ as

$$a_C\,(\mathsf{n}\,(\mathsf{b}\, w\, s), (F\, s) \circ e_{(\mathsf{b}\, w\, s)}).$$

To see that it is extensional, let $s, s' : \mathsf{ImS}\, w$ and $\sigma : s \approx s'$. It is enough to show that there is $\alpha : \mathsf{n}\,(\mathsf{b}\, w\, s) \approx \mathsf{n}\,(\mathsf{b}\, w\, s')$ such that $F\, s\, b \approx F\, s'(B_\alpha\, b)$ for all $b : B\,(\mathsf{n}\,(\mathsf{b}\, w\, s))$. We may take $\alpha := \mathsf{n} \restriction ((\mathsf{b}\, w \restriction)\sigma)$, then the second equality follows from coherence of F.

Let now γ, F, F' be as above, and $\phi : F \approx_\gamma F'$, $s : \mathsf{ImS}\, w$. Applying extensionality of a_C, it is enough to show

$$(\mathsf{n}\,(\mathsf{b}\, w\, s), (F\, s) \circ e_{(\mathsf{b}\, w\, s)}) \approx (\mathsf{n}\,(\mathsf{b}\, w'\,(\mathsf{ImS}_\gamma\, s)), F'\,(\mathsf{ImS}_\gamma\, s) \circ e_{(\mathsf{b}\, w'\,(\mathsf{ImS}_\gamma\, s))}).$$

For the first component we may take $\alpha := \mathsf{n} \restriction (\mathsf{extb}\, \gamma)$. It remains to show that, for every $s : \mathsf{ImS}$,

$$F\, s \approx_{(\mathsf{ImS}\,(\mathsf{n}\,(\mathsf{b}\, w\, s)) \Rightarrow C)} (F'\,(\mathsf{ImS}_\gamma\, s)) \circ \mathsf{ImS}_{(\mathsf{extb}\, \gamma)}.$$

For this it is enough to use ϕ. $\square$

3.3. The type of recursively defined maps.

We would like apply this construction recursively on $w : W$, in order to get a term in $\prod_w \mathsf{ImS}\, w \Rightarrow C$. In order to do this we need to make sure that coherence is preserved. However, the scope of the coherence condition is limited to functions defined on immediate subtrees of a given tree. We could say that coherence has a local character, as opposed to the global character of the commutativity condition $a_C \circ \mathsf{P}_B\, h \approx h \circ \mathsf{s}$ which applies to functions defined on any tree. This makes the coherence condition not suited to be carried along through recursion. However, let us call an extensional function $k : \mathsf{ImS}\, w \Rightarrow C$ *recursively defined* if it is obtained applying recst to a coherent family F and each of the functions $F\, s$ is also recursively defined. Since the functions $F\, s$ are over an immediate subtree, we may construct the type of all inductively defined functions on immediate subtrees using dependent W-types.

For $w : W$ and $k : \mathsf{ImS}\, w \Rightarrow C$, let $\mathsf{RecDef}\, w\, k$ be the dependent W-type on

$$I := \sum_{w:W} \mathsf{ImS}\, w \Rightarrow C, \qquad X\,(w, k) := \sum_{F:\mathsf{CohMaps}\, w} k \approx \mathsf{recst}\, w\, F,$$

$$Y\,(w, k)\,(F, \_) := \mathsf{ImS}\, w, \qquad d\,(w, k_w)\,(F, \_)\, s := (\mathsf{b}\, w\, s, F\, s).$$

From its names and branching functions, we obtain the following terms:

$$\mathsf{rdfam}\, w\, k : \mathsf{RecDef}\, w\, k \to \mathsf{CohMaps}\, w$$

$$\mathsf{rdeq}\, w\, k : \prod_{(D:\mathsf{RecDef}\, w\, k)} k \approx \mathsf{recst}\, w\, (\mathsf{rdfam}\, w\, k\, D),$$

$$\mathsf{rdcond}\, w\, k : \prod_{(D:\mathsf{RecDef}\, w\, k)} \prod_{(s:\mathsf{ImS}\, w)} \mathsf{RecDef}\, (\mathsf{b}\, w\, s)\, (\mathsf{rdfam}\, w\, k\, D\, s).$$

Recursively defined map are stable under transport.

Lemma 3.10. *Let* $\gamma : w \approx_W w'$ *and* $k : \mathsf{ImS}\, w \Rightarrow C$. *Then*

$$\mathsf{RecDef}\, w\, k \to \mathsf{RecDef}\, w'\, (k \circ \mathsf{ImS}_{\gamma^{-1}}).$$

Proof. This is proven by induction on $D : \mathsf{RecDef}\, w\, k$ into the type

$$\prod_{(w':W)} \prod_{(\gamma:w\approx w')} \mathsf{RecDef}\, w'\, (k \circ \mathsf{ImS}_{\gamma^{-1}}).$$

Since k is obtained applying recst to $F := \mathsf{rdfam}\, w\, k\, D$, Lemma 3.9 implies

$$k \circ \mathsf{ImS}_{\gamma^{-1}} \approx \mathsf{recst}\, w'\, (\mathsf{CohMaps}_\gamma\, F).$$

In order to apply dsup it only remains to provide a branching function, that is, to show that for each $s' : \mathsf{ImS}\, w'$, $\mathsf{CohMaps}_\gamma\, F\, s'$ is recursively defined. This is precisely the inductive hypothesis applied to $(\mathsf{extb}\, \gamma^{-1}\, s')^{-1} : \mathsf{b}\, w\, (B_{\mathsf{nf}^\mathsf{r}\gamma^{-1}}\, s') \approx \mathsf{b}\, w'\, s'$ and $\mathsf{rdcond}\, w\, k\, D$. $\qquad\square$

As it may be expected, recursively defined maps are unique.

Lemma 3.11. *Let* $w : W$, $k : \mathsf{ImS}\, w \Rightarrow C$ *and* $k' : \mathsf{ImS}\, w \Rightarrow C$. *Then*

$$\mathsf{RecDef}\, w\, k \to \mathsf{RecDef}\, w\, k' \to k \approx k'.$$

Proof. This is proven by induction on $D : \mathsf{RecDef}\, w\, k$. It is enough to show that

$$\mathsf{recst}\, w\, (\mathsf{rdfam}\, w\, k\, D) \approx \mathsf{recst}\, w\, (\mathsf{rdfam}\, w\, k'\, D').$$

Using $\mathsf{extrecst}$ from Lemma 3.9 this reduces to show that, for every $s : \mathsf{ImS}\, w$

$$\mathsf{rdfam}\, w\, k\, D\, s \approx \mathsf{rdfam}\, w\, k'\, D'\, s.$$

But this is precisely the inductive hypothesis. $\qquad\square$

Proposition 3.12. *Let* $\gamma : w \approx_W w'$, $k : \mathsf{ImS}\, w \Rightarrow C$ *and* $k' : \mathsf{ImS}\, w' \Rightarrow C$. *Then*

$$\mathsf{RecDef}\, w\, k \to \mathsf{RecDef}\, w'\, k' \to k \approx k' \circ \mathsf{ImS}_\gamma.$$

Proof. Straightforward from the previous two Lemmas. $\qquad\square$

We are now able to construct a family of extensional functions on immediate subtrees

Proposition 3.13. *For every* $w : W$ *there are*

$$\mathsf{recImS}\, w : \mathsf{ImS}\, w \Rightarrow C \qquad and \qquad \mathsf{recImSpf}\, w : \mathsf{RecDef}\, w\, (\mathsf{recImS}\, w).$$

Proof. The proof is by W-elimination on $w : \mathsf{W}_{A_0, B_0}$ into the type

$$\mathsf{W}^{\mathsf{per}}_B \, w \, w \; \longrightarrow \; \sum_{k:\mathsf{ImS}\,w \Rightarrow C} \mathsf{RecDef} \, w \, k.$$

To simplify the exposition, we may assume without loss of generality that all the trees we shall be dealing with are extensional. This is the case since all (immediate) subtrees of an extensional tree are also extensional, as one can immediately see recalling that a tree is extensional if and only if the branching function is extensional as a function $B \, (\mathsf{b} \, w) \Rightarrow W$.

We have an extensional tree of the form $w \equiv \mathsf{sup} \, a \, f$, and the inductive hypothesis consists of a family

$$IH : \prod_{(s:\mathsf{ImS}\,w)} \sum_{(k:\mathsf{ImS}\,(f\,s) \Rightarrow C)} \mathsf{RecDef} \, (f \, s) \, k.$$

We work towards applying Lemma 3.9. A coherent family is given by

$$F := \mathrm{pr}_1 \circ IH : \prod_{s:\mathsf{ImS}\,w} \mathsf{ImS} \, (f \, s) \Rightarrow C.$$

Its coherence follows from Proposition 3.12 using $\mathrm{pr}_2 \circ IH : \prod_{(s:\mathsf{ImS}\,w)} \mathsf{RecDef} \, (f \, s) \, (F \, s)$.

We thus may define

$$\mathsf{recImS} \, w := \mathsf{recst} \, w \, F : \mathsf{ImS} \, w \Rightarrow C \qquad \text{and}$$

$$\mathsf{recImSpf} \, w := \mathsf{dsup} \, \_\, (F, \rho \, \_) \, (\mathrm{pr}_2 \circ IH) : \mathsf{RecDef} \, w \, (\mathsf{recImS} \, w). \qquad \square$$

3.4. **Characterisation of algebra morphisms.** Before proceeding to construct the universal arrow and show its commutativity and uniqueness, we provide in Theorem 3.17 a characterisation of algebra morphisms as those maps which are recursively defined.

Consider the setoid $\mathsf{Alg}_B(\mathsf{s}, a_C)$ of algebra morphisms from s to a_C:

$$\sum_{h:W \Rightarrow C} h \circ \mathsf{s} \approx a_C \circ (\mathsf{P}_B \, h) \qquad \text{and} \qquad (h, \_) \approx (h', \_) := h \approx h',$$

and the setoid of families of recursively defined maps RFam:

$$\sum \left(F : \prod_w \mathsf{ImS} \, w \Rightarrow C \right) \prod_w \mathsf{RecDef} \, w \, (Fw) \qquad \text{and} \qquad (F, \_) \approx (F', \_) := \prod_w F \, w \approx F' \, w.$$

We shall prove that these setoids are isomorphic. Since any two families of recursively defined maps are equal because of Lemma 3.11, the family recImS from Proposition 3.13 is the only inhabitant of RFam. From this observation and the isomorphism above, initiality will follow immediately.

Given any function $h : W \Rightarrow C$, we may consider the family of restrictions $h|_w := h \circ m_w : \mathsf{ImS} \, w \Rightarrow C$.

Lemma 3.14. *Let $h : W \Rightarrow C$ be an algebra morphism. For every $w : W$, $h|_w$ is recursively defined.*

Proof. The proof is by W-elimination on w into $\mathsf{RecDef} \, w \, h|_w$. Let then $w \equiv \mathsf{sup} \, a \, f$. In order to apply dsup, we first need to provide a coherent family $F : \mathsf{CohMaps} \, w$ and a proof that $h|_w \approx \mathsf{recst} \, w \, F$.

The family consists of the restrictions of h to subtrees of w: $F \, s := h \circ m_{(f\,s)}$. Its coherence follows easily from extensionality of h and extb. For $s : \mathsf{ImS} \, w$ we have:

$$h|_w \, s \approx a_C \circ (\mathsf{P}_B \, h) \circ \mathsf{us} \, (f \, s)$$
$$\approx a_C \, (\mathsf{n} \, (f \, s), h \circ \mathsf{b} \, (f \, s))$$
$$\approx a_C \, (\mathsf{n} \, (f \, s), h|_{(f\,s)} \circ e_{(f\,s)})$$
$$\approx \mathsf{recst} \, w \, (\lambda s. h|_{(f\,s)}) \, s.$$

The inductive hypothesis witnesses that each of the restrictions $h|_{(f\,s)}$ is recursively defined. $\qquad\square$

Hence there is a function, obviously extensional,

$$\mathsf{rest} : \mathsf{Alg}_B(\mathsf{s}, a_C) \Rightarrow \mathsf{RFam}$$

that maps an algebra morphism into its family of restrictions.

In order to construct an inverse, let us say that a family $F : \prod_w \mathsf{ImS}\,w \Rightarrow C$ is *coherent* if, for every $\gamma : w \approx w'$, $F\,w \approx (F\,w') \circ \mathsf{ImS}_\gamma$. Any such family gives rise to an extensional function $\mathsf{cmprh}\,F : W \Rightarrow \mathsf{P}_B\,C$ defined by

$$w : W \longmapsto (\mathsf{n}\,w : A,\ (F\,w) \circ e_w : B\,(\mathsf{n}\,w) \Rightarrow C).$$

Extensionality is immediate using $\mathsf{n}\,\vec{\mathsf{r}}$ and coherence of F. Further, using reflexivity on $\mathsf{n}\,w$ and proof-irrelevance of the setoid family $\lambda a.B\,a \Rightarrow C$ we obtain a term

$$\mathsf{extcmprh} : \Big(\prod_{w:W} F\,w \approx F'\,w \Big) \longrightarrow \mathsf{cmprh}\,F \approx \mathsf{cmprh}F'.$$

It follows from Proposition 3.12 that families in RFam are coherent, hence the assignment $RF \mapsto a_C \circ (\mathsf{cmprh}\,(\mathrm{pr}_1\,RF))$ gives rise to an extensional function which, abusing notation, we shall denote

$$\lambda F.a_C \circ (\mathsf{cmprh}\,F) : \mathsf{RFam} \Rightarrow (W \Rightarrow C).$$

Extensionality for this function is obtained from $\mathsf{extcmprh}$ and $a_C\,\vec{\mathsf{r}}$. The next two results show that $a_C \circ (\mathsf{cmprh}\,F)$ is in fact an algebra morphism.

Lemma 3.15. *For every $F : \mathsf{RFam}$ and every $w : W$, the function $(a_C \circ (\mathsf{cmprh}\,F))|_w : \mathsf{ImS}\,w \Rightarrow C$ is recursively defined.*

Proof. In order to apply dsup we first need a coherent family $G : \mathsf{CohMaps}\,w$ and a proof of $(a_C \circ (\mathsf{cmprh}\,F))|_w \approx \mathsf{recst}\,w\,G$. We may take $G\,s := F\,(\mathsf{b}\,w\,s)$, coherence follows as before from Proposition 3.12 and the fact that each $F\,w'$ is recursively defined. The equation is just a matter of unfolding the definitions: on each $s : \mathsf{ImS}\,w$ the two sides are judgementally equal. Since every function in the family F is recursively defined, we obtain a canonical witness of $\mathsf{RecDef}\,w\,(a_C \circ (\mathsf{cmprh}\,F))$. $\qquad\square$

Corollary 3.16. *For every $F : \mathsf{RFam}$*

$$\mathsf{cmprh}\,F \approx (\mathsf{P}_B\,(a_C \circ (\mathsf{cmprh}\,F))) \circ \mathsf{us}.$$

Hence the function $a_C \circ (\mathsf{cmprh}\,F) : W \Rightarrow C$ is an algebra morphism.

Proof. For $w : W$ we have

$$\begin{aligned}
\mathsf{cmprh}\,F\,w &\approx (\mathsf{n}\,w, (F\,w) \circ e_w) \\
&\approx (\mathsf{n}\,w, a_C \circ (\mathsf{cmprh}\,F) \circ m_w \circ e_w) \\
&\approx \mathsf{P}_B\,(a_C \circ (\mathsf{cmprh}\,F))\,(\mathsf{n}\,w, \mathsf{b}\,w),
\end{aligned}$$

where the second equality uses proof-irrelevance of the setoid family $\lambda a.B\,a \Rightarrow C$ and the fact just proved that both functions are recursively defined on w, and so equal by Lemma 3.11. $\qquad\square$

We are now able to state our characterisation of algebra morphisms as families of recursively defined maps.

Theorem 3.17. *The two functions*

$$\mathsf{Alg}_B(\mathsf{s}, a_C) \mathrel{\substack{\xrightarrow{\text{rest}} \\ \xleftarrow[a_C \circ (\mathsf{cmprh\_})]{}}} \mathsf{RFam}$$

are inverse to each other: for every $h : \mathsf{Alg}_B(\mathsf{s}, a_C)$ *and* $F : \mathsf{RFam}$

$$\prod_w \mathsf{rest}\,(a_C \circ (\mathsf{cmprh}\,F))\,w \approx F\,w \qquad and \qquad a_C \circ (\mathsf{cmprh}\,(\mathsf{rest}\,h)) \approx h.$$

Proof. The first equality follows immediately from Lemmas 3.11 and 3.15. The second one follows from the fact that h is an algebra morphism once we observe that, for every $w : W$,

$$a_C \circ (\mathsf{cmprh}\,(\mathsf{rest}\,h)))\,w \equiv a_C\,(\mathsf{n}\,w, h|_w \circ e_w)$$
$$\equiv a_C \circ (\mathsf{P}_B\,h) \circ \mathsf{us}\,w. \qquad \square$$

3.5. **Initiality of s.** Since any two families of recursively defined maps are equal because of Lemma 3.11, the family recImS from Proposition 3.13 is the only inhabitant of RFam. From this observation and the characterisation above, initiality of $\mathsf{s} : \mathsf{P}_B\,W \Rightarrow W$ follows immediately. Unfolding the construction we see that the unique algebra morphism is given by

$$a_C \circ (\mathsf{cmprh}\,\mathsf{recImS}) : W \Rightarrow C,$$

where we omitted the proof term witnessing that the maps in recImS are recursively defined. Corollary 3.16 implies that it is an algebra morphism and uniqueness follows from the fact that $\mathsf{Alg}_B(\mathsf{s}, a_C)$ is isomorphic to the unit type. We have proven the following.

Theorem 3.18. *The category of setoids* Std *has initial algebras for polynomial endofunctors, that is, the following type is inhabited:*

$$\prod_{\left(\substack{A:\mathsf{Std} \\ B:\mathsf{Fam}\,A}\right)} \prod_{\left(\substack{C:\mathsf{Std} \\ a_C:\mathsf{P}_B\,C \Rightarrow C}\right)} \sum_{(h:\mathsf{Alg}_B(\mathsf{s},a_C))} \prod_{(h':\mathsf{Alg}_B(\mathsf{s},a_C))} h \approx h'.$$

As already mentioned, a theorem by Gambino and Hyland ensures that a locally cartesian closed category has initial algebras for dependent polynomial endofunctors as soon as it has initial algebras for polynomial endofunctors [17]. Formulating this result for e-categories in Martin-Löf type theory should pose no conceptual problems. Once this is done, Theorem 3.18 would yield as a corollary that Std has dependent W-types as soon as the underlying universe in Martin-Löf type theory has dependent W-types.

In order to generalise the argument presented in this paper to general quotient completions, it may be better to first consider the notion of *homotopy exact completion* introduced by van den Berg and Moerdijk [9]. Indeed, such completion is probably the closest to the setoid construction described here among all quotient completions mentioned in the introduction. The first step in that direction would consist of defining a category-theoretic version of dependent W-types in intensional type theory as certain homotopy-initial algebra, along the lines of the work done in Homotopy Type Theory by Awodey, Gambino and Sojakova in [3]. Once this is done, we expect our argument to carry over without major obstacles.

References

[1] P. Aczel. The type theoretic interpretation of constructive set theory. In A. MacIntyre, L. Pacholski, and J. Paris, editors, *Logic Colloquium '77*, volume 96 of *Studies in Logic and the Foundations of Mathematics*, pages 55–66. North-Holland, Amsterdam, 1978.

[2] P. Aczel. The type theoretic interpretation of constructive set theory: Inductive definitions. In Ruth Barcan Marcus, Georg J.W. Dorn, and Paul Weingartner, editors, *Logic, Methodology and Philosophy of Science VII*, volume 114 of *Studies in Logic and the Foundations of Mathematics*, pages 17–49. Elsevier, 1986.

[3] S. Awodey, N. Gambino, and K. Sojakova. Homotopy-initial algebras in type theory. *Journal of the ACM*, 63(6):51:1–51:45, 2017.

[4] G. Barthe, V. Capretta, and O. Pons. Setoids in type theory. *J. Funct. Program.*, 13(2):261–293, 2003.

[5] B. van den Berg. Inductive types and exact completion. *Annals of Pure and Applied Logic*, 134(2):95–121, 2005.

[6] B. van den Berg and I. Moerdijk. Aspects of predicative algebraic set theory I: Exact completion. *Annals of Pure and Applied Logic*, 156(1):123–159, 2008.

[7] B. van den Berg and I. Moerdijk. Aspects of predicative algebraic set theory, II: Realizability. *Theoretical Computer Science*, 412(20):1916–1940, 2011. Girard's Festschrift.

[8] B. van den Berg and I. Moerdijk. Aspects of predicative algebraic set theory III: Sheaves. *Proceedings of the London Mathematical Society*, 105(5):1076–1122, 2012.

[9] B. van den Berg and I. Moerdijk. Exact completion of path categories and algebraic set theory. Part I: Exact completion of path categories. *Journal of Pure and Applied Algebra*, 222(10):3137–3181, 2018.

[10] L. Birkedal, A. Carboni, G. Rosolini, and D. S. Scott. Type theory via exact categories (extended abstract). In *Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science LICS '98*, pages 188–198. IEEE Computer Society Press, 1998.

[11] L. Bressan. An extension of the Minimalist Foundation. Master's thesis, Università degli Studi di Padova, 2015.

[12] A. Carboni and E.M. Vitale. Regular and exact completions. *Journal of Pure and Applied Algebra*, 125(1-3):79–116, 1998.

[13] P. Dybjer. Inductively defined sets in Martin-Löf's type theory. In *Proceedings of the Workshop on General Logic, Edinburgh, February 1987*. Number ECS-LFCS-88-52 in LFCS Report Series, 1988.

[14] P. Dybjer. Representing inductively defined sets by wellorderings in Martin-Löf's type theory. *Theoretical Computer Science*, 176(1):329–335, 1997.

[15] J. Emmenegger and E. Palmgren. Exact completion and constructive theories of sets. Submitted. arXiv:1710.10685, 2017.

[16] N. Gambino. Presheaf models for constructive set theories. In L. Crosilla and P. Schuster, editors, *From Sets and Types to Topology and Analysis: Towards Practicable Foundations for Constructive Mathematics*, pages 62–77. Oxford University Press, 2005.

[17] N. Gambino and M. Hyland. Wellfounded trees and dependent polynomial functors. In *Types for proofs and programs*, volume 3085 of *Lecture Notes in Computer Science*, pages 210–225. Springer, 2004.

[18] N. Gambino and J. Kock. Polynomial functors and polynomial monads. *Mathematical Proceedings of the Cambridge Philosophical Society*, 154(1):153–192, 2013.

[19] M. Hofmann. On the interpretation of type theory in locally cartesian closed categories. In *Proceedings of Computer Science Logic, Lecture Notes in Computer Science*, pages 427–441. Springer, 1994.

[20] M. Hofmann. *Extensional Constructs in Intensional Type Theory*. Distinguished dissertations in Computer Science. Springer, 1997.

[21] M.E. Maietti. A minimalist two-level foundation for constructive mathematics. *Annals of Pure and Applied Logic*, 160(3):319–354, 2009.

[22] M.E. Maietti and G. Rosolini. Quotient completion for the foundation of constructive mathematics. *Logica Universalis*, 7(3):371–402, 2013.

[23] M.E. Maietti and G. Rosolini. Relating quotient completions via categorical logic. In P. Schuster and D Probst, editors, *Concepts of Proof in Mathematics, Philosophy, and Computer Science*, pages 229–250. De Gruyter, 2016.

[24] P. Martin-Löf. Constructive mathematics and computer programming. 104, 12 1982.

[25] P. Martin-Löf. *Intuitionistic type theory. Notes by G. Sambin of a series of lectures given in Padua, June 1980*. Studies in Proof Theory. Bibliopolis, Napoli, 1984.

[26] I. Moerdijk and E. Palmgren. Wellfounded trees in categories. *Annals of Pure and Applied Logic*, 104(1):189–218, 2000.

[27] I. Moerdijk and E. Palmgren. Type theories, toposes and constructive set theory: predicative aspects of AST. *Annals of Pure and Applied Logic*, 114(1):155 – 201, 2002. Troelstra Festschrift.

[28] B. Nordström, K. Petersson, and J.M. Smith. *Programming in Martin-Löf's type theory. An introduction.* Oxford University Press, Oxford, 1990.

[29] E. Palmgren. Type-theoretic interpretation of iterated, strictly positive inductive definitions. *Archive for Mathematical Logic*, 32(2):75–99, 1992.

[30] E. Palmgren and O. Wilander. Constructing categories and setoids of setoids in type theory. *Logical Methods in Computer Science*, 10(3), 2014.

[31] K. Petersson and D. Synek. A set constructor for inductive sets in Martin-Löf's type theory. In *Proceedings of the 1989 Conference on Category Theory and Computer Science, Manchester, U.K.*, volume 389 of *Lecture Notes in Computer Science*, pages 128–140. Springer-Verlag, 1989.

[32] R. A. G. Seely. Locally cartesian closed categories and type theory. *Mathematical Proceedings of the Cambridge Philosophical Society*, 95(1):33–48, 1984.

[33] A. Swan. W-types with reductions and the small object argument. arXiv:1802.07588, 2018.