Logical Methods in Computer Science
Volume 17, Issue 3, 2021, pp. 28:1–28:33
https://lmcs.episciences.org/

Submitted Sep. 16, 2019
Published Sep. 24, 2021

W-TYPES IN SETOIDS

JACOPO EMMENEGGER

Matematiska institutionen, Stockholms unversitet, Sweden.
Current address: DIMA, Università degli Studi di Genova, 16146 Genova, Italy.
e-mail address: emmenegger@dima.unige.it

Abstract. We present a construction of W-types in the setoid model of extensional Martin-Löf type theory using dependent W-types in the underlying intensional theory. More precisely, we prove that the internal category of setoids has initial algebras for polynomial endofunctors. In particular, we characterise the setoid of algebra morphisms from the initial algebra to a given algebra as a setoid on a dependent W-type. We conclude by discussing the case of free setoids. We work in a fully intensional theory and, in fact, we assume identity types only when discussing free setoids. By using dependent W-types we can also avoid elimination into a type universe. The results have been verified in Coq and a formalisation is available on the author's GitHub page.

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 [ML84, ML82] produces an inductive type whose terms can be understood as well-founded trees. We provide a construction, verified in Coq, of W-types in the setoid model of extensional Martin-Löf type theory using dependent W-types in the underlying intensional theory. Although we work internally in intensional Martin-Löf type theory, we present our results using the category-theoretic language. More precisely, we consider a logical-framework presentation of a dependent type theory with unit type, $\sum$-types, $\prod$-types and dependent W-types and we show in Theorem 4.20 that the internal category of setoids has initial algebras for polynomial endofunctors. These were identified as a category-theoretic counterpart of W-types by Moerdijk and Palmgren [MP00]. Initiality of an algebra and the induction principle of the underlying type are related via the notion of contractibility in [AGS17].

Dependent W-type were introduced by Petersson and Synek [PS89], see also [NPS90, Chapter 16], to provide a constructor for general inductive data types, and are also known as indexed W-types or tree types. The dependent W-type constructor produces a family of mutually inductive types, as opposed to the single inductive type produced by the ordinary W-type constructor. Indeed, ordinary W-types are type-theoretically equivalent

Key words and phrases: W-types, polynomial functors, Martin-Löf type theory, setoids.
This work was partially funded by EPSRC grant EP/T000252/1.

DOI:10.46298/LMCS-17(3:28)2021

to dependent W-types indexed over the unit type. Gambino and Hyland [GH04], see also [GK13], formulated a dependent version of polynomial endofunctors and identified in their initial algebras a category-theoretic counterpart of dependent W-types.

By setoid we mean a type equipped with a type-valued total equivalence relation. That these form a model of extensional Martin-Löf type theory that interprets most type constructors has been known for some time. Most type constructors, as only recently Palmgren [Pal19] has provided us with a solution to the long standing problem of interpreting a type universe, and in fact a whole hierarchy, in the setoid model. As to W-types, it was again Palmgren who first showed how to construct W-types for setoids in intensional Martin-Löf type theory, see [Bre15]. His approach uses extensively the elimination principle of W-types into a type universe. The novelty of our approach consists in the use of dependent W-types instead.

We use dependent W-types in two crucial steps. First, to define the (partial) equivalence relation on a W-type that gives rise to the initial algebra setoid W. Second, to characterise the setoid of morphisms of algebras from the (candidate) initial algebra to a given algebra. After these steps, it only takes an easy induction to conclude initiality of W. To briefly describe the steps where dependent W-types are used, let us recall that we may regard terms of a W-type as well-founded trees: a canonical tree is recursively specified providing its root and its immediate subtrees, *i.e.* other well-founded trees that are to be connected to the root.

The relation that we define in 3.1 stipulates that two trees are matching if their roots are equal and their immediate subtrees on equal branches are matching, where by equal we mean the setoid equality. By using dependent W-types, this inductively defines a partial relation on the underlying W-type, and we say that a tree is *extensional* if it is matching with itself. Palmgren's construction of W-types for setoids, as well as the other constructions that we discuss below, use the same relation, but constructed in different ways. The initial algebra will be the setoid of extensional trees with algebra map given by the constructor of the W-type.

Given an algebra a on a setoid A, we then need to construct an algebra morphism f from W to A, that is, a function commuting with the algebra maps, and to show that it is the unique such. Commutativity tells us, roughly, that the action of f on an extensional tree w is determined, via the algebra a, by the action of f on the immediate subtrees of w. By using dependent W-types, we make precise in Definition 4.7 what it means for a function on extensional trees to be determined by its action on immediate subtrees, and call these functions *telescopic*. Actually, for the construction itself we find it more convenient to speak of telescopic functions on immediate subtrees rather than on trees, but here we can safely ignore the difference. In Theorem 4.18 we prove that a function on extensional trees is an algebra morphism if and only if it is telescopic.

The characterisation in Theorem 4.18 allows us to reduce the problem of finding a unique algebra morphism, to the problem of finding a unique telescopic function. To this aim, and thanks to the inductive nature of telescopic functions, we directly use the elimination principle of dependent W-types. In this sense, we believe that Theorem 4.18 makes explicit, in the case of setoids, the connection between the commutativity condition for an algebra morphism out of the initial algebra, and the inductive definition of the morphism itself.

In particular, a common aspect of arguments that construct W-types is the use of the set, or setoid, of all subtrees of a tree, usually obtained as the transitive closure of the immediate subtree relation. This set may have a complicated construction in categories

and in intensional type theories, but it is important for inductive arguments. We can avoid dealing with transitive closures when proving Theorem 4.18 since the definition of telescopic functions only involves the setoid of immediate subtrees of a tree, which has a natural and straightforward definition as an image factorisation, see Definition 3.7 and Remark 3.8.

We also use dependent W-types to compare the setoid of extensional trees with respect to discrete setoids to the discrete setoid on the W-type of the underlying types. In particular, we see in Corollary 5.9 that every discrete tree with nodes from a 0-type, *i.e.* a type with decidable equality, is extensional. It seems that, without further assumptions, not every discrete tree is extensional. However, if function extensionality holds, then it is possible to identify the setoid of extensional trees on discrete setoids with a subsetoid of discrete trees, see Theorem 5.8.

The setoid construction that we consider is an instance of a quotient completion, see [MR13, MR16]. The author is aware of two other constructions of W-types for quotient completions. The first one, that we already mentioned, was formulated by Palmgren for the setoid model in intensional Martin-Löf type theory and then adapted by Bressan [Bre15] to minimalist type theory [Mai09]. The argument requires a 'large' elimination principle for W-types, in the sense that it must be possible to eliminate into a type universe or a universe of propositions. The second construction is due to van den Berg [Ber05] and it applies to exact completions of categories with finite limits. In intensional Martin-Löf type theory, the assumption of finite limits is met by the category of types assuming function extensionality and Uniqueness of Identity Proofs, by its full subcategory on the 0-types assuming only function extensionality [UFP13, RS15], and by the e-category of types assuming only UIP [EP20]. However, there is little hope that an internal category of types in a fully intensional type theory would have ordinary finite limits. Setoids are also closely related to groupoids. Vidmar has given a construction of initial algebras for polynomial endofunctors on groupoids from initial algebras for polynomial endofunctors on sets [Vid18].

In the next section we discuss the preliminaries needed for the construction. Section 3 contains the construction of the algebra of extensional trees and the definition of the setoid family of immediate subtrees. The proof of its initiality is in Section 4. We conclude the paper with a discussion of the case of extensional trees over discrete setoids in Section 5. Each section begins with a brief overview of the content.

The Coq formalisation of the present paper is available on the author's GitHub page [Emm18] and it includes Definitions 2.2 and 2.3 and all numbered definitions and results in Sections 2.5 and 3 to 5, except for Proposition 2.9, a Coq proof of which can be found at [Pal12a].

2. Preliminaries

We present the type theory we will be working with in Section 2.1; briefly review W-types in Section 2.2 and their dependent verion in Section 2.3; and recall some basic facts and definitions about setoid and setoid families in Section 2.4, and about polynomial functors in Section 2.5.

2.1. **The type-theoretic setting.**

We work in a logical framework formulation of Martin-Löf type theory, see [NPS90, Part III], similar to the fragment of Coq that we used for the formalisation. The logical framework consists of record types, $\prod$-types and a universe à la Russell U, which is the type Set in the Coq formalisation. Logic is interpreted according

to the propositions-as-types interpretation. When we wish to emphasise that a type is in U we say that it is a *small type*. Judgemental equality is denoted $\equiv$ and application of function terms will usually be denoted by concatenation as in $f\,b$. However, in the presence of multiple arguments we may adopt the notation with parenthesis as in $ar(i, n, b)$.

The theory itself is then specified by declaring constants and equations. Specifically, we require the universe U to be closed under $\prod$-types, and to contain the unit type $\mathsf{1}$, $\sum$-types, and dependent W-types. Ordinary W-types are recovered as dependent W-types over the unit type $\mathsf{1}$, but in the Coq formalisation these are assumed to be primitive for convenience. In Figures 1 and 2 we spell out the constants and equations for (dependent) W-types in the form of rules to increase readability. In these rules, the premises of W-FORM and DW-FORM are left understood in the other rules. We might as well drop (some) subscripts if they are clear from the context. The system has η-conversion only for $\prod$-types.

Note that we do not require identity types. These will be assumed only in Section 5 for discussing the case of discrete setoids. It is well-known that record types can be replaced with $\sum$-types at the expense of readability. In this case we would not need to assume that these coincide with the $\sum$-types in the theory.

2.2. **W-types.** W-types can be used to construct several inductive types, including natural numbers and lists [Dyb97], and to give a constructive justifications of certain theories of inductive definitions [Pal92]. More generally, W-types provide a predicative counterpart to the notion of well-ordering. Furthermore, they are instrumental in Aczel's model of Constructive Zermelo-Fraenkel set theory [Acz78], in the form of the type of iterative sets, where they also allow to interpret the Regular Extension Axiom which adds general inductive definitions to CZF [Acz86]. The idea of a type of iterative sets is also central in Palmgren's interpretation of type universes in the setoid model [Pal19].

By elimination of W-types, there are function terms

$$\mathsf{node} : \mathsf{W}_B \to A \qquad \text{and} \qquad \mathsf{ist} : \prod_{w:\mathsf{W}_B} B\,(\mathsf{node}\,w) \to \mathsf{W}_B$$

Figure 1: Rules for W-types.

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

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

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

Figure 2: Rules for dependent W-types.

$$\frac{I : \mathsf{U} \qquad N : I \to \mathsf{U} \qquad Br : \prod_{i:I} N\, i \to \mathsf{U} \qquad ar : \prod_{(i:I)} \prod_{(n:N\, i)} Br(i,n) \to I}{\mathsf{DW}_{Br,ar} : I \to \mathsf{U}}\ \textsf{DW-FORM}$$

$$\frac{i : I \qquad n : N\, i \qquad f : \prod_{b:Br(i,n)} \mathsf{DW}_{Br,ar}\, ar(i,n,b)}{\mathsf{dsup}_{Br,ar}(i,n,f) : \mathsf{DW}_{Br,ar}\, i}\ \textsf{DW-INTRO}$$

$$\frac{\begin{array}{l} C : \prod_{i:I} \mathsf{DW}_{Br,ar}\, i \to \mathsf{U} \\ c : \prod_{(i:I)} \prod_{(n:N\, i)} \prod_{\left(f:\prod_{b:Br(i,n)} \mathsf{DW}_{Br,ar}\, ar(i,n,b)\right)} \left(\prod_{b:Br(i,n)} C(ar(i,n,b), f\, b) \right) \to C(i, \mathsf{dsup}_{Br,ar}(i,n,f)) \end{array}}{\mathsf{rec}^{\mathsf{DW}}_{Br,ar,C,c} : \prod_{i:I} \prod_{w:\mathsf{DW}_{Br,ar}\, i} C(i,w)}\ \textsf{DW-ELIM}$$

$$\frac{i : I \qquad n : N\, i \qquad f : \prod_{b:Br(i,n)} \mathsf{DW}_{Br,ar}\, (ar(i,n,b))}{\mathsf{rec}^{\mathsf{DW}}_{Br,ar,C,c}(i, \mathsf{dsup}_{Br,ar}(i,n,f)) \equiv c(i,n,f,\lambda b.\mathsf{rec}^{\mathsf{DW}}_{C,c}(ar(i,n,b), f\, b))}\ \textsf{DW-CONV}$$

such that $\mathsf{node}\,(\mathsf{sup}\, a\, f) \equiv a$ and $\mathsf{ist}\,(\mathsf{sup}\, a\, f) \equiv f$. Henceforth, we write ist_w for $\mathsf{ist}\, w$.

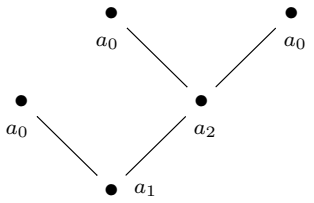
We refer to terms of a W-type W_B as *trees*, to terms $a : A$ as *nodes*, to terms $b : B\, a$ as *branches* and, for every branch $b : B\, a$, to the term $\mathsf{ist}_w\, b$ as the *immediate subtree of* w *on branch* b.

In the set-theoretic interpretation of type theory, W-types are sets of well-founded trees with labelled nodes. The set A is the set of labels for the nodes and, 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 $f\colon B_a \to \mathsf{W}_B$ which stipulate that the tree $f(b)$ is attached to the branch $b \in B_a$.

Figure 3 shows a tree in W_B when A contains three elements a_0, a_1 and a_2 such that B_{a_0} is empty and $B_{a_1} = \{0,1\} = B_{a_2}$. The corresponding canonical element of W_B is displayed on the right-hand side together with the five functions mapping a branch to the tree attached to it, among which occur the empty function $f_\emptyset$ into W_B.

As it is well-known, these sets of trees can be characterised more abstractly as free term algebras for infinitary single-sorted signatures. In this view, the set A contains the function symbols of the signature, while (the cardinality of) $B\, a$ is the arity of symbol a. Terms are built out of function symbols according to composition instructions specified

Figure 3: Canonical term of a W-type.



$$\mathsf{sup}\, a_1 \begin{pmatrix} 0 \mapsto \mathsf{sup}\, a_0\, f_\emptyset \\ 1 \mapsto \mathsf{sup}\, a_2 \begin{pmatrix} 0 \mapsto \mathsf{sup}\, a_0\, f_\emptyset \\ 1 \mapsto \mathsf{sup}\, a_0\, f_\emptyset \end{pmatrix} \end{pmatrix}$$

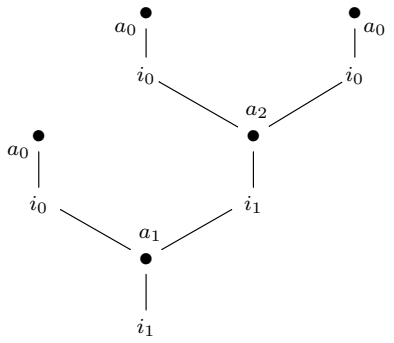
by functions $f\colon B_a \to \mathsf{W}_B$ which stipulate that, for $b \in B_a$, the term $f(b)$ occurs as b-th argument. The tree in Figure 3 is the term also written $a_1(a_0, a_2(a_0, a_0))$, in a signature with function symbols a_0 with arity 0 and a_1, a_2 with arity 2.

2.3. **Dependent W-types.** In the same vein, dependent W-types can be understood as free term algebras for infinitary *multi-sorted* signatures: sorts are elements of I, elements of N_i are function symbols with codomain sort i and the arity of a function symbol $a \in N_i$ is given by the function $ar_{i,a} : B_{i,a} \to I$ that maps each b-th argument of a to its sort.

Figure 4 depicts an element of a dependent W-type. Compared to the graphical representation of trees in Figure 3, the sort i of a function symbol $a \in N_i$ appears as an additional leg pointing downwards out of the node corresponding to a, and each branch $b \in Br_{i,a}$ is labelled with the sort $ar(i, a, b) \in I$. The dependent tree in Figure 4 is a term in the signature containing two sorts i_0 and i_1, and three function symbols, a_0 of sort i_0 and a_1, a_2 of sort i_1, such that a_0 has empty arity, a_1 has arity i_0 and i_1, and a_2 has arity i_0 and i_0. This is formally specified by requiring that $i_0, i_1 \in I$, $a_0 \in N_{i_0}$, $a_1, a_2 \in N_{i_1}$, $Br_{i_0,a_0} = \emptyset$, $Br_{i_1,a_1} = \{0, 1\} = Br_{i_1,a_2}$, $ar(i_0, a_0)$ is the empty function into I, $ar(i_1, a_1)$ maps 0 to i_0 and 1 to i_1 and $ar(i_1, a_2)$ maps 0 and 1 to i_0. A common presentation of the term in Figure 4 is $a_1(a_0, a_2(a_0, a_0)) : i_1$.

Note that there must be at least one node with no branches, *i.e.* at least one constant, in order for these trees to be well-founded. Indeed, in terms of signatures, these sets of trees consist of closed terms. This is reflected in type theory: whenever B is a family of non-empty types, W_B is type-theoretically equivalent to the empty type $\mathsf{0}$, when the latter

Figure 4: Canonical term of a dependent W-type.



$$\mathsf{dsup}\, i_1\, a_1 \begin{pmatrix} 0 \mapsto \mathsf{dsup}\, i_0\, a_0\, f_\emptyset \\ 1 \mapsto \mathsf{dsup}\, i_1\, a_2 \begin{pmatrix} 0 \mapsto \mathsf{dsup}\, i_0\, a_0\, f_\emptyset \\ 1 \mapsto \mathsf{dsup}\, i_0\, a_0\, f_\emptyset \end{pmatrix} \end{pmatrix}$$

is available. In order to make this precise, the meaning of 'non-empty' can be understood either as $\prod_{a:A}(B\,a \to \mathsf{0}) \to \mathsf{0}$ or as $\prod_{a:A} ||B\,a||$ if propositional truncation is available, in the sense that assuming either of them proves that W_B is equivalent to $\mathsf{0}$.

Similarly to the non-dependent case, using the elimination rule of dependent W-types, we obtain for every $i : I$ two function terms

$$\mathsf{dnode}_i : \mathsf{DW}\,i \to N\,i \tag{2.1}$$

$$\mathsf{dist}_i : \prod_{(w:\mathsf{DW}\,i)} \prod_{(b:Br(i,\mathsf{dnode}_i\,w))} \mathsf{DW}\,ar(i, \mathsf{dnode}_i\,w, b), \tag{2.2}$$

such that $\mathsf{dnode}_i\,\mathsf{dsup}(i, n, f) \equiv n$ and $\mathsf{dist}_i\,\mathsf{dsup}(i, n, f) \equiv f$. We refer to I as the type of *indices* or sorts, to N and Br as the *node family* and the *branching family*, respectively, and to $ar : \prod_i \prod_n Br(i, n) \to I$ as the *arity function*.

The following examples are just for the sake of illustrating the use of dependent W-types and we do not use the definitions therein elsewhere in the paper.

Examples 2.1.

(1) Let $X : \mathsf{U}$ and $Y : X \to \mathsf{U}$. We can define a type of (root-to-leaf) paths in a tree $w : \mathsf{W}_Y$ as a dependent W-type over W_Y as follows. Say that a path in $w \equiv \mathsf{sup}\,x\,f$ is a branch $y : Y\,x$ together with a path in the immediate subtree $f\,y$.

The family Path_Y of paths in trees in W_Y is the dependent W-type on W_Y with node and branching families

$$N \coloneqq \lambda w.Y\,(\mathsf{node}\,w) : \mathsf{W}_Y \to \mathsf{U}, \qquad Br \coloneqq \lambda w, y.1 : \prod_{w:\mathsf{W}_Y} N\,w \to \mathsf{U},$$

and with arity function $ar \coloneqq \lambda w, y, \_.\mathsf{ist}_w\,y : \prod_w \prod_y \prod_{\mathsf{1}} \mathsf{W}_Y$. A canonical element of $\mathsf{Path}_Y\,w$ consists of a branch y of w together with a function $\mathsf{1} \to \mathsf{Path}_Y(\mathsf{ist}_w\,y)$, *i.e.* a path in the immediate subtree $\mathsf{ist}_w\,y$.

(2) If the theory has finite sums (in fact, the type $\mathsf{2}$ suffices) and the empty type $\mathsf{0}$, it is possible to define a type of finite paths in a tree $w : \mathsf{W}_Y$ similarly to the previous example. Say that a finite path in $w \equiv \mathsf{sup}\,x\,f$ is either the empty path, or it is a branch $y : Y\,x$ together with a finite path in the immediate subtree $f\,y : \mathsf{W}_Y$.

The family $\mathsf{FinPath}_Y : \mathsf{W}_Y \to \mathsf{U}$ of finite paths in trees in W_Y is the dependent W-type on W_Y with node and branching families

$$N \coloneqq \lambda w.Y\,(\mathsf{node}\,w) + \mathsf{1}, \qquad Br \coloneqq \lambda w, z. \begin{cases} z \equiv \mathsf{inl}\,y & \mapsto \mathsf{1} \\ z \equiv \mathsf{inr}\,t & \mapsto \mathsf{0} \end{cases}$$

and with arity function $ar \coloneqq \lambda w, z, \_. \begin{cases} z \equiv \mathsf{inl}\,y & \mapsto \mathsf{ist}_w\,y \\ z \equiv \mathsf{inr}\,t & \mapsto R_{\mathsf{0}} \end{cases} : \prod_w \prod_z \prod \mathsf{W}_Y$ where $R_{\mathsf{0}} : \mathsf{0} \to \mathsf{W}_Y$ is the eliminator of the empty type.

A canonical element of $\mathsf{FinPath}_Y\,w$ consists either of a term $t : \mathsf{1}$ and a function $\mathsf{0} \to \mathsf{W}_Y$, which we regard as 'the' empty path, or of a branch b of w and a function $\mathsf{1} \to \mathsf{FinPaths}_Y\,(\mathsf{ist}_w\,y)$, *i.e.* a finite path in the immediate subtree $\mathsf{ist}_w\,y$. Either this path consists of a term of $\mathsf{1}$ and an empty function, in which case the path is over with length one. Or it consists of a branch y' and a finite path in $\mathsf{ist}_{\mathsf{ist}\,y}\,y'$.

The function term $\mathsf{FinPath}_Y\,w \to \mathsf{N}$ that maps each finite path to its length can be easily defined using the elimination rule of dependent W-types in Figure 2.

2.4. **Setoids and setoid families.** 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 0-types and mere relations. See [BCP03] for a survey of possible definitions. In our context we define a setoid as follows.

Definition 2.2. A *setoid* A is a list $(|A|, \approx_A, r_A, s_A, t_A)$ where $|A| : \mathsf{U}$ is a small type, the type family $\approx_A$: $|A| \to |A| \to \mathsf{U}$ is an equivalence relation with proofs of reflexivity, symmetry and transitivity given by, respectively:

$$r_A : \prod_{a:|A|} a \approx_A a, \qquad s_A : \prod_{a,a':|A|} a \approx_A a' \to a' \approx_A a,$$

$$t_A : \prod_{a,a',a'':|A|} a \approx_A a' \to a' \approx_A a'' \to a \approx_A a''.$$

The *type of setoids* Std is defined as a record on the types of $|A|, \approx_A, r_A, s_A$ and t_A.

For a setoid A, we refer to the relation $\approx_A$ as the *equality of* A.

Since our theory has a unit type, every small type X gives rise to a setoid on X with equality $\lambda a, a'.1$, called the the *codiscrete setoid on* X.

In Section 5 we will have identity types available. In this case every small type X can be equipped with the equality $\lambda x, x'.x =_X x'$ given by the identity type. This setoid is called the *discrete setoid on* X. More generally, a setoid A is *discrete* or (*free*) if $a \approx_A a'$ if and only if $a =_{|A|} a'$ for every $a, a' : A$.

In the category of setoids that we will define below, the codiscrete setoid over any inhabited type is a terminal object, and it is isomorphic to the discrete setoid on $\mathsf{1}$ if identity types are available.

Definition 2.3. Let A and B be setoids. A function term $f : |A| \to |B|$ is *extensional (with respect to* $\approx_A$ *and* $\approx_B$*)* if there is a term of type

$$\mathsf{ext}\, f \coloneqq \prod_{a,a':|A|} a \approx_A a' \longrightarrow f\, a \approx_B f\, a'. \tag{2.3}$$

We will refer to extensional function terms as extensional functions, or simply as functions, and we will write $|f|$ for the underlying function term of an extensional function f.

The setoid $A \Rightarrow B$ of *extensional functions* from A to B has the type of extensional functions

$$\sum_{f:|A|\to|B|} \mathsf{ext}\, f$$

as underlying type and the equivalence relation

$$f \approx_{A\Rightarrow B} g \coloneqq \prod_{a:|A|} |f|\, a \approx_B |g|\, a$$

as equality.

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

Remark 2.4. Both type families ext and $\approx_{A \Rightarrow B}$ defined in 2.3 are instances of a more general binary relation on the type $|A| \to |B|$, namely

$$\mathsf{extgen}(f, g) \coloneqq \prod_{a,a' : |A|} a \approx_A a' \longrightarrow f\, a \approx_B g\, a'.$$

It is easy to see that this is a partial equivalence relation on $|A| \to |B|$ whose domain, *i.e.* the type of function terms f such that $\mathsf{extgen}(f, f)$, is the type of extensional function $|A \Rightarrow B|$. Furthermore, two extensional functions f and g are equal if and only if the type $\mathsf{extgen}(|f|, |g|)$ is inhabited.

In the rest of the paper we write $a : A$ for a setoid A, to mean $a : |A|$, and we often not distinguish between an extensional function $f : A \Rightarrow B$ and its underlying function term $|f| : |A| \to |B|$. We also write $f \mathbin{\Rsh} \alpha : f\, a \approx_B f\, a'$ for the proof of extensionality of f applied to $\alpha : a \approx_A a'$. Occasionally, we 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.

Setoids provide us with a formulation of the notion of category without equality on objects that was introduced by Aczel, who formalised it in Lego [Acz93]. See also [HS00] for a formalisation in Coq that uses setoids with Prop-valued equalities.

Definition 2.5. A *(locally small) e-category* $\mathbb{A}$ consists of a type of objects $\mathrm{Obj}_{\mathbb{A}}$ and a family of setoids $\mathrm{Hom}_{\mathbb{A}} : \mathrm{Obj}_{\mathbb{A}} \to \mathrm{Obj}_{\mathbb{A}} \to \mathsf{Std}$, of arrows, together with function terms for identity and composition, where the latter is extensional, in the sense that it 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 the family of setoids $\mathrm{Hom}_{\mathbb{A}}$.

An *e-functor* between two e-categories $\mathbb{A}$ and $\mathbb{B}$ consists of a function term $F : \mathrm{Obj}_{\mathbb{A}} \to \mathrm{Obj}_{\mathbb{B}}$ together with a term of type

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

and proof terms witnessing that it is functorial with respect to the equalities of the family of setoids $\mathrm{Hom}_{\mathbb{B}}$.

An *e-natural transformation* between two e-functors $F, G \colon \mathbb{A} \to \mathbb{B}$ consists of a term

$$n : \prod_{a : \mathrm{Obj}_{\mathbb{A}}} \mathrm{Hom}_{\mathbb{B}}(F\, a, G\, a)$$

whose action on $a : \mathrm{Obj}_{\mathbb{A}}$ we denote as n_a, together with a term of type

$$\prod_{a,a' : \mathrm{Obj}_{\mathbb{A}}} \prod_{f : a \Rightarrow a'} G\, f \circ n_a \approx n_{a'} \circ F\, f,$$

where $\approx$ is the equality of $\mathrm{Hom}_{\mathbb{B}}(F\, a, G\, a')$.

The names 'e-category', 'e-functor', etc. were introduced to distinguish these concepts from the standard ones. However, since these are the only formulations that appear in the present paper, we will just say 'category' to mean 'e-category' and similarly for the others.

For a setoid A, the *discrete category* $A^{\#}$ on A is defined as follows. Its type of objects is $|A|$ 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 $\approx_A$ is symmetric, the category $A^{\#}$ is a groupoid, *i.e.* a category where every arrow is invertible.

Functors between two categories $\mathbb{A}$ and $\mathbb{B}$ and natural transformations between them form a category $\mathsf{Fun}(\mathbb{A}, \mathbb{B})$. We denote the action of a functor F on an arrow $\alpha : \mathrm{Hom}_{\mathbb{A}}(a, a')$ as $F_\alpha : \mathrm{Hom}_{\mathbb{B}}(F\,a, F\,a')$.

Setoids and extensional functions form a category [PW14].

Definition 2.6. The *category of setoids* is defined as follows. The type of objects is the type of setoids Std and the family of arrows is the family $\lambda A, B.A \Rightarrow B$ of setoids of extensional functions. Identity and composition are defined in the obvious way, the latter will be denoted as $g \circ f$. We abuse notation and denoted this category also as Std.

For a setoid A, we define

$$\mathsf{Fam}\,A \coloneqq \mathsf{Fun}(A^{\#}, \mathsf{Std})$$

and call a functor $B : \mathsf{Fam}\,A$ a *setoid family over* A. We refer to the action of B on an arrow $\alpha : a \approx_A a'$ of $A^{\#}$ as *transport along* α. For $b : B\,a$ and $b' : B\,a'$ we define

$$b \approx_\alpha b' \;\coloneqq\; B_\alpha\,b \approx_{B\,a'} b'. \tag{2.4}$$

For every extensional function $f : A' \Rightarrow A$ and every setoid family $B : \mathsf{Fam}\,A$, there is a setoid family $f^*B : \mathsf{Fam}\,A'$ defined by $(f^*B)\,a \coloneqq B\,(f\,a)$ and $(f^*B)_\alpha \coloneqq B_{(f\,\mathrel{\Vdash}\,\alpha)}$. This action extends to a functor from $\mathsf{Fam}\,A$ to $\mathsf{Fam}\,A'$.

Remark 2.7. Besides the standard functoriality conditions on identities and composites, setoid families enjoy an additional coherence. Indeed, for every $\alpha, \alpha' : a \approx_A a'$, it is $\alpha \approx \alpha'$ as arrows in $A^{\#}$. It follows by extensionality of B on arrows that

$$B_\alpha \approx_{(B\,a \Rightarrow B\,a')} B_{\alpha'}$$

for every α and α' as above. Indeed, setoid families in this sense are the proof-irrelevant setoid families studied in [Pal12b]. In particular, for every $\alpha, \alpha' : a \approx_A a'$, $b : B\,a$ and $b' : B\,a'$, it is

$$b \approx_\alpha b' \longleftrightarrow b \approx_{\alpha'} b' \tag{2.5}$$

and, for every $\alpha : a \approx_A a$ and $b, b' : B\,a$, it is

$$b \approx_{B\,a} b' \longleftrightarrow b \approx_\alpha b'. \tag{2.6}$$

In fact, setoid families are formalised in [Emm18] directly as a record on the types of function terms and their functoriality and proof-irrelevance properties.

It is an important feature of the category of sets in set theory that the slice over a set A is equivalent to families of sets indexed over A. The same holds for the category of setoids in the fragment of intensional Martin-Löf type theory that we are considering, see Section 2.1.

Theorem 2.8. *For a setoid A, the slice category Std/A is equivalent to the category of setoid families $\mathsf{Fam}\,A$.*

Proof. We describe the two functors, leaving the straightforward verification of the equivalence to the reader.

Every setoid family $B : \mathsf{Fam}\,A$ gives rise to a setoid $(\sum_{a:A} B\,a, \approx)$ where the equality is

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

The projection into A is extensional, with extensionality proof also given by the first projection. This defines an extensional function into A.

A natural transformation n from B to C is mapped to the extensional function with underlying term $\lambda(a,b).(a, n_a\, b)$. Its extensionality follows, for every $(\alpha,\beta) : (a,b) \approx (a',b')$, from

$$C_\alpha(n_a\, b) \approx_{C\, a'} n_{a'}\, (B_\alpha b) \approx_{C\, a'} n_{a'}\, b'$$

which holds by naturality of n and extensionality of $n_{a'}$.

Conversely, let $f : B \Rightarrow A$ be a function. Consider, for every $a : A$, the subsetoid of B on those b such that $f\, b \approx_A a$, *i.e.* the type

$$\sum_{b:B} f\, b \approx_A a$$

with equality $(b,\_) \approx (b',\_) \coloneqq b \approx_B b'$. This assignment extends to a setoid family whose transport along $\alpha : a \approx_A a'$ maps a pair (b,β) to the pair (b,β'), where $\beta' : f\, b \approx_A a'$ is the concatenation of β and α.

An arrow in Std/A from f to some $g : C \Rightarrow A$ is a function $k : B \Rightarrow C$ such that $g \circ k \approx f$. This gives rise to a natural transformation as follows. For $a : A$, the underlying function term maps (b,β) to $(k\, b, \beta')$, where $\beta' : g\, (k\, b) \approx_A a$ is the concatenation of $\beta : f\, b \approx_A a$ with the proof of $g \circ k \approx f$ applied to b. This function term is clearly extensional as k is. As transport along $\alpha : a \approx_A a'$ does not modify the first component, naturality in $a : A$ is trivial. □

2.5. **Polynomial functors and W-types.** Recall that a category is *locally cartesian closed* if it has a terminal object and all its slices are cartesian closed, that is, have all binary products and exponentials. Theorem 2.8 may be used to prove the following result, another proof can be found in [EP20].

Proposition 2.9. *The category of setoids* Std *is locally cartesian closed.*

In a locally cartesian closed category $\mathbb{C}$, for every arrow $f\colon B \to A$, the pullback functor f^* has a left adjoint $f_!$, defined by post-composition with f, and a right adjoint f_*, as below.

$$\mathbb{C}/B \underset{f_!}{\overset{f_*}{\underset{\longleftarrow f^* \longrightarrow}{\overset{\longleftarrow}{\longrightarrow}}}} \mathbb{C}/A \tag{2.7}$$

When f is the unique arrow $B \to 1$ to the terminal object, the above functors become

$$\mathbb{C}/B \underset{B_!}{\overset{(-)^B}{\underset{\longleftarrow (-)\times B \longrightarrow}{\overset{\longleftarrow}{\longrightarrow}}}} \mathbb{C}.$$

It follows that every arrow $f\colon B \to A$ in $\mathbb{C}$ gives rise to a functor $P_f\colon \mathbb{C} \to \mathbb{C}$, the *polynomial endofunctor associated to* f, defined as the composite

$$\mathbb{C} \xrightarrow{(-)\times B} \mathbb{C}/B \xrightarrow{f_*} \mathbb{C}/A \xrightarrow{A_!} \mathbb{C}. \tag{2.8}$$

A *polynomial endofunctor* is then defined to be an endofunctor on $\mathbb{C}$ which is naturally isomorphic to P_f for some f in $\mathbb{C}$.

An *algebra for a polynomial endofunctor* P is given by an object X and an arrow $s\colon PX \to X$, called *algebra map*. Such an algebra is *initial* if for any algebra $t\colon PY \to Y$, there is $h\colon X \to Y$ such that $t \circ (P\,h) = h \circ s$ and h is the unique such. 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 category of small types and function terms is locally cartesian closed if the universe has $\mathsf{1}$, identity types, $\sum$ 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 type family $f^{-1} \coloneqq \lambda a.\, \sum_{b:B} f(b) =_A a$, with algebra map defined by

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

It is not difficult to see that initiality of $(\mathsf{W}_{f^{-1}}, \mathsf{sup})$ is logically equivalent to the recursion principle of $\mathsf{W}_{f^{-1}}$ [MP00].

In Std, we take advantage of Theorem 2.8 and formulate the notion of polynomial functor for setoid families.

Definition 2.10. Let $B : \mathsf{Fam}\, A$ be a setoid family over a setoid A. The *polynomial endofunctor associated to* B is the functor $\mathsf{Std} \to \mathsf{Std}$ defined on $X : \mathsf{Std}$ by

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

and on $f : X \Rightarrow Y$ by

$$(\mathsf{P}_B f)(a, k) \coloneqq (a, f \circ k).$$

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

In fact, the formalisation [Emm18] only contains the definition of the action of P_B on setoids and functions.

We need to make sure that this definition does coincide with the standard one in (2.8) for an arbitrary locally cartesian closed category.

Let us begin by unfolding the definition of P_f in (2.8) for an extensional function $f : B \Rightarrow A$. Recall that the three adjoint functors in (2.7) are constructed using $\sum$-types and $\prod$-types. More precisely, the first functor in (2.8) maps a setoid X to the projection $\mathrm{pr}_2 : X \times B \Rightarrow B$, and the last one maps a function $x : X \Rightarrow A$ to its domain X. To describe the action of the functor $f_* : \mathsf{Std}/B \to \mathsf{Std}/A$ on a function $y : Y \Rightarrow B$, let us say that a pair (a, g) in

$$\sum_{a:A} S(f)\, a \Rightarrow Y$$

is a *local section of* y if, for every $(b, \_) : S(f)\, a$, it is $y\,(g\,(b, \_)) \approx_B b$. Two local sections (a, g) and (a', g') are equal if there is $\alpha : a \approx_A a'$ such that $g \approx g' \circ S(f)_\alpha$. This defines a setoid LS_y of local sections of y, and the function $f_*\, y : LS_y \Rightarrow A$ is the obvious projection.

Let us now denote by $S(f)$ the setoid family constructed from the extensional function f in the proof of Theorem 2.8. Unfolding the definitions, one sees that, for every X, the setoid $P_f X$ is in bijection with $\mathsf{P}_{S(f)} X$, and that the bijection is natural in X. If we denote by E the inverse construction to S from the same proof, it also follows that for every setoid family $B : \mathsf{Fam}\, A$ the functor $P_{E(B)}$ is naturally isomorphic to P_B.

Definition 2.11. Let $B : \mathsf{Fam}\, A$ be a setoid family.

(1) An *algebra for* P_B *in* Std is a setoid X together with an extensional function $\mathsf{P}_B X \Rightarrow X$.
(2) Let $x : \mathsf{P}_B X \Rightarrow X$ and $y : \mathsf{P}_B Y \Rightarrow Y$ be algebras for P_B. The *setoid* $\mathsf{Alg}(x, y)$ *of algebra morphisms from* x *to* y consists of functions $h : X \Rightarrow Y$ such that $h \circ x \approx y \circ \mathsf{P}_B h$. In formulas:

$$|\mathsf{Alg}(x, y)| \coloneqq \sum_{h:W \Rightarrow C} h \circ x \approx y \circ (\mathsf{P}_B h)$$

and $(h, \_) \approx_{\mathsf{Alg}(x,y)} (h', \_) \coloneqq h \approx h'$.

Definition 2.12. Let $B : \mathsf{Fam}\, A$ be a setoid family and C a setoid. A family of extensional functions $F : \prod_{x:|A|} B\, x \Rightarrow C$ is *coherent with respect to* B if, for every $a, a' : A$, if $\alpha : a \approx_A a'$ then $F\, a \approx F\, a' \circ B_\alpha$.

Let the type

$$\mathsf{isCoh}_B\, F \coloneqq \prod_{(a,a':A)} \prod_{(\alpha:a \approx a')} F\, a \approx F\, a' \circ B_\alpha$$

be the type of proofs that F is coherent.

Lemma 2.13. *Let* $f : A' \Rightarrow A$ *be an extensional function and let* $B : \mathsf{Fam}\, A$ *be a setoid family over* A*. For every family* $F : \prod_{x:|A'|} B\,(f\, x) \Rightarrow C$ *of extensional functions, the following are equivalent.*

(1) *The family* F *is coherent with respect to* $f^* B : \mathsf{Fam}\, A'$*.*
(2) *The function term* $\lambda x.(f\, x, F\, x) : A' \to \mathsf{P}_B C$ *is extensional.*

Proof. Let $\alpha' : a \approx_{A'} a'$. Then $(f\, a, F\, a) \approx (f\, a', F\, a')$ if and only if there is $\alpha : f\, a \approx_A f\, a'$ such that $F\, a \approx F\, a' \circ B_\alpha$. Thus it clearly follows from coherence of F taking $\alpha \coloneqq f \mathbin{\Rsh} \alpha'$. The converse is a direct consequence of proof irrelevance of B, see (2.5) in Remark 2.7. □

Remark 2.14. The value of the polynomial functor P_B on a setoid C can be described as the total setoid of the setoid family $\lambda x.B\, x \Rightarrow C$ over A. Accordingly, Definition 2.12 and Lemma 2.13 can be phrased (and proved) in greater generality. However we do not need that generality here.

3. The algebra of extensional trees

Let $B : \mathsf{Fam}\, A$ be a setoid family over a setoid A. In this section we construct an algebra from the W-type $\mathsf{W}_{|B|}$ on the underlying type family $|B| : |A| \to \mathsf{U}$. We keep A and B fixed throughout the section and we drop the subscript B in the polynomial functor P_B.

The underlying setoid of the algebra is constructed in Section 3.1. Before showing that sup lifts to an algebra map in Section 3.3, we introduce the setoid family of immediate subtrees in Section 3.2. Section 3.3 contains also a proof that the algebra map is invertible.

3.1. **The setoid of extensional trees.** To construct an algebra, we first need a setoid of trees, that is, an equivalence relation on $\mathsf{W}_{|B|}$. However, we should not expect $\mathsf{W}_{|B|}$ to be the underlying type of our setoid of trees though or, in other words, we should not expect our relation to be total on $\mathsf{W}_{|B|}$. Consider in analogy the situation described in Remark 2.4 with function terms: the domain of the equivalence relation extgen on the function type consists precisely of the extensional functions.

In order to find sufficient conditions for defining a suitable relation, let us suppose that such a partial equivalence relation $\approx_W \colon \mathsf{W}_{|B|} \to \mathsf{W}_{|B|} \to \mathsf{U}$ does exist. This means that, if we denote by $W : \mathsf{Std}$ the setoid induced by $\approx_W$ on its domain $|W| \coloneqq \sum_{w:\mathsf{W}_{|B|}} w \approx_W w$, there is an initial algebra $s : \mathsf{P}W \Rightarrow W$ whose underlying function term $|s| : (\sum_{a:A} B\,a \Rightarrow W) \to |W|$ is a lift of $\mathsf{sup} : \left(\sum_{a:A} |B|\,a \to \mathsf{W}_{|B|}\right) \to \mathsf{W}_{|B|}$. Since s must be a bijection by Lambek's Theorem, for every tree $w \equiv \mathsf{sup}\,a\,f$ such that $w \approx_W w$ there is $g : B\,a \Rightarrow W$ such that $s(a, g) \approx_W w$. This can be equivalently stated as follows.

$$\text{If } \mathsf{sup}\,a\,f \approx_W \mathsf{sup}\,a\,f, \text{ then for every } b, b' : B\,a, \text{ if } b \approx_{B\,a} b' \text{ then } f\,b \approx_W f\,b'. \tag{W1}$$

Equivalently, for every tree w in the domain of $\approx_W$ the immediate-subtree function term ist_w is extensional. Second, again because s is a bijection, and by definition of P in 2.10, it must be

$$\mathsf{sup}\,a\,f \approx_W \mathsf{sup}\,a'\,f' \quad \text{if and only if} \quad \begin{cases} \text{there is } \alpha : a \approx_A a' \text{ such that} \\ f\,b \approx_W f'\,(B_\alpha\,b) \text{ for every } b : B\,a. \end{cases} \tag{W2}$$

Let us now try to use (W1-2) to define $\approx_W$. In condition (W2) the relation $\approx_W$ occurs on the right-hand side only between immediate subtrees. This feature makes (W2) a possible candidate for the inductive step in an inductive definition of the relation $\approx_W$. However, it does not seem possible to derive (W1) from (W2). Instead, we combine them in the following definition. Note that in 3.1.2 below the equality between b and b' is over the term α as in (2.4).

Definition 3.1. Let $a, a' : A$, $f : |B|\,a \to \mathsf{W}_{|B|}$ and $f' : |B|\,a' \to \mathsf{W}_{|B|}$. Two trees $w \equiv \mathsf{sup}\,a\,f$ and $w' \equiv \mathsf{sup}\,a'\,f'$ are *matching* if

(1) the nodes are equal, *i.e.* there is $\alpha : a \approx_A a'$, and
(2) for every $b : B\,a$ and $b' : B\,a'$, if $b \approx_\alpha b'$ then the immediate subtrees $f\,b$ and $f'\,b'$ are matching.

A tree that is matching itself will be called *self-matching* or *extensional*.

Accordingly, we define the type family $\mathsf{W}^{\mathsf{per}}_B : \mathsf{W}_{|B|} \to \mathsf{W}_{|B|} \to \mathsf{U}$ of proofs that two trees are matching as the (curried version of) the dependent W-type on $\mathsf{W}_{|B|} \times \mathsf{W}_{|B|}$ with family of nodes

$$N \coloneqq \lambda(w, w').\mathsf{node}\,w \approx_A \mathsf{node}\,w' : \mathsf{W}_{|B|} \times \mathsf{W}_{|B|} \to \mathsf{U}$$

branching family

$$Br \coloneqq \lambda(w, w'), \alpha. \sum_{b,b'} b \approx_\alpha b' : \prod_I N\,(w, w') \to \mathsf{U}$$

and arity function

$$ar \coloneqq \lambda(w, w'), \alpha, (b, b', \beta).(\mathsf{ist}_w\,b, \mathsf{ist}_{w'}\,b') : \prod_I \prod_N Br((w, w'), \alpha) \to I.$$

The matching relation satisfies (W1) and (W2): for the former take α in 3.1.1 to be reflexivity on a, and for the latter take $b' \coloneqq B_\alpha\, b$ and reflexivity on b' in 3.1.2.

Remark 3.2. For two trees $w, w' : \mathsf{W}_{|B|}$, a proof term of 3.1.1 is

$$\alpha : \mathsf{node}\, w \approx_A \mathsf{node}\, w'$$

and a proof term of 3.1.2 is

$$\phi : \prod_{b:B\,(\mathsf{node}\, w)} \prod_{b':B\,(\mathsf{node}\, w')} \Big(b \approx_\alpha b' \longrightarrow \mathsf{W}^{\mathsf{per}}_B\, (\mathsf{ist}_w\, b)\, (\mathsf{ist}_{w'}\, b')\Big).$$

If we have two such terms, then a proof that w and w' are matching is

$$\mathsf{dsup}\, (w, w')\, \alpha\, \phi : \mathsf{W}^{\mathsf{per}}_B\, w\, w'.$$

Conversely, if w and w' are matching with proof γ, then α is obtained by applying the dependent node function $\mathsf{dnode}_{(w,w')}$ from (2.1) to γ, and ϕ by evaluating the dependent branching function $\mathsf{dist}_{(w,w')}$ from (2.2) on γ.

Proposition 3.3. *The type family* $\mathsf{W}^{\mathsf{per}}_B : \mathsf{W}_{|B|} \to \mathsf{W}_{|B|} \to \mathsf{U}$ *is a partial equivalence relation on* $\mathsf{W}_{|B|}$*, that is, the following types are inhabited:*

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

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

Proof. The proof terms are obtained from straightforward applications of the elimination rule for dependent W-types. Alternatively, one may use Remark 3.2 and recursion on $\mathsf{W}_{|B|}$. □

We can now form the setoid of extensional trees.

Definition 3.4. Define the *setoid* W_B *of extensional trees* as the pair $(|W_B|, \approx_{W_B}) : \mathsf{Std}$, where $|W_B|$ is the type of extensional (or self-matching) trees, and two terms in $|W_B|$ are equal if their underlying trees are matching. In formulas:

$$|W_B| \coloneqq \sum_{w:\mathsf{W}_{|B|}} \mathsf{W}^{\mathsf{per}}_B\, w\, w \qquad \text{and} \qquad (w, \_) \approx_{W_B} (w', \_) \coloneqq \mathsf{W}^{\mathsf{per}}_B\, w\, w'.$$

We often leave the proof term in $\mathsf{W}^{\mathsf{per}}_B\, w\, w$ implicit, drop the subscript B when it is clear from context, and write $w : W$ to mean $w : \mathsf{W}_{|B|}$ and w extensional.

Similarly to the situation described in Remark 2.4, to prove that a tree w is extensional it is sufficient to consider an instance of the type of ϕ in Remark 3.2, namely the one where α is reflexivity at the node of w.

Lemma 3.5. *A tree* $w \equiv \mathsf{sup}\, a\, f$ *is extensional if and only if for every* $b, b' : B\, a$*, if* $b \approx b'$ *then the trees* $f\, b$ *and* $f\, b'$ *are matching.*

In particular, every immediate subtree of an extensional tree is extensional and f *lifts to an extensional function* $B\, a \Rightarrow W$*.*

Proof. If w is extensional the conclusion follows by (W1). The converse follows from Remark 3.2 choosing reflexivity as α. □

The function term $\mathsf{node} : \mathsf{W}_{|B|} \to |A|$ lifts to an extensional function $\mathsf{node} : W \Rightarrow A$, with proof of extensionality given by α in Remark 3.2. By Lemma 3.5, for every extensional tree w the function term $\mathsf{ist}_w : |B|\,(\mathsf{node}\,w) \to \mathsf{W}_{|B|}$ lifts to an extensional function $\mathsf{ist}_w : B\,(\mathsf{node}\,w) \Rightarrow W$. So we have a family

$$\mathsf{ist} : \prod_{w:W} B\,(\mathsf{node}\,w) \Rightarrow W \tag{3.1}$$

of extensional functions. We continue writing ist_w for $\mathsf{ist}\,w$ and we will do the same also for others dependent function terms on W. As the subscript will always be a tree in W, no confusion should arise with the action of setoid families on equalities.

Also to prove that two *extensional* trees are matching it is enough to consider just an instance of the type of ϕ in Remark 3.2, as for extensional functions in Remark 2.4.

Lemma 3.6. *Two extensional trees $w \equiv \mathsf{sup}\,a\,f$ and $w' \equiv \mathsf{sup}\,a'\,f'$ are matching if and only if there is $\alpha : a \approx_A a'$ such that $f \approx f' \circ B_\alpha$.*

Equivalently, for every $w, w' : W$, it is $w \approx_W w'$ if and only if there are

$$\alpha : \mathsf{node}\,w \approx_A \mathsf{node}\,w' \qquad \textit{and} \qquad \psi : \mathsf{ist}_w \approx \mathsf{ist}_{w'} \circ B_\alpha.$$

Proof. If w and w' are matching, the conclusion follows by (W2). Conversely, by Remark 3.2 it is enough to construct a term

$$\phi : \prod_{b,b'} b \approx_\alpha b' \to f\,b \approx_W f'\,b'.$$

By Lemma 3.5 we may assume that the function f' is extensional. It follows that for every $b : B\,a$, $b' : B\,a'$, if $\beta : b \approx_\alpha b'$, then $f\,b \approx_W f'\,b'$ using $\psi\,b : f\,b \approx_W f'\,(B_\alpha\,b)$ and $f' \mathrel{\Rsh} \beta$. □

3.2. **Setoid families on extensional trees.** Extensionality of $\mathsf{node} : W \Rightarrow A$ yields, for every $\gamma : w \approx_W w'$ between extensional tree, a function

$$B_{\mathsf{node} \mathrel{\Rsh} \gamma} : B\,(\mathsf{node}\,w) \Rightarrow B\,(\mathsf{node}\,w') \tag{3.2}$$

where $\mathsf{node} \mathrel{\Rsh} \gamma$ denotes the proof of extensionality (2.3) of node applied to γ. Since B is a setoid family, the family of setoids $\lambda w.B\,(\mathsf{node}\,w) : W \to \mathsf{Std}$ is a setoid family with the functions in (3.2) as transports. We refer to this setoid family as the family of *branches of trees*.

There is also another natural setoid family on extensional trees which will turn out to be instrumental in the characterisation of algebra morphisms in Section 4.

Definition 3.7. Let $w \equiv \mathsf{sup}\,a\,f$ be an extensional tree. An *immediate subtree of w* is a subtree index $b : B\,a$, and two immediate subtrees b and b' of w are equal if the corresponding trees $f\,b$ and $f\,b'$ are matching.

Accordingly, the setoid of *immediate subtrees of w*, denoted $\mathsf{ISTrees}\,w$, has $|B|\,(\mathsf{node}\,w)$ as underlying type, and

$$s \approx_{\mathsf{ISTrees}\,w} s' \coloneqq \mathsf{ist}_w\,s \approx_W \mathsf{ist}_w\,s'$$

as equality.

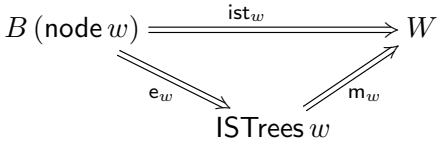
For $s : \mathsf{ISTrees}\,w$ and $\gamma : w \approx_W w'$, the assignment

$$\mathsf{ISTrees}_\gamma\,s \coloneqq B_{\mathsf{node} \mathrel{\Rsh} \gamma}\,s : \mathsf{ISTrees}\,w'$$

defines transport maps for $\mathsf{ISTrees}$. Hence we obtain a setoid family $\mathsf{ISTrees} : \mathsf{Fam}\,W$, the *family of immediate subtrees*.

Remark 3.8. The category of setoids Std is exact and, assuming the identity type, it is the exact completion of its subcategory of discrete setoids. We refer to [EP20] for details but see also [BM18]. Being exact, every function has a factorisation as a regular epi followed by a mono, called image factorisation. Being an exact completion, every function $f : X \Rightarrow Y$ has a canonical such factorisation. Briefly: the setoid Z has $|X|$ as underlying type and $\lambda x, x'.\, (f\, x \approx_Y f\, x')$ as equality; the regular epi $X \Rightarrow Z$ has $\lambda x.x$ as underlying function term, which is extensional because f is; and the mono $Z \Rightarrow Y$ has $|f|$ as underlying function term, which is extensional by definition of Z. In fact, this canonical factorisation exists also when identity types are not assumed.

By applying this factorisation to the function $\mathsf{ist}_w : B\,(\mathsf{node}\, w) \Rightarrow W$ we recover the setoid of immediate subtrees of w:



where $\mathsf{e}_w \coloneqq (\mathsf{id}, \mathsf{ist}_w \Rsh)$ and $\mathsf{m}_w \coloneqq (|\mathsf{ist}_w|, \mathsf{id})$ denote the regular epi and the mono, respectively, arising from the factorisation of ist_w.

These form in turn two families of extensional functions

$$\mathsf{e} : \prod_{w:W} B\,(\mathsf{node}\, w) \Rightarrow \mathsf{ISTrees}\, w \qquad \text{and} \qquad \mathsf{m} : \prod_{w:W} \mathsf{ISTrees}\, w \Rightarrow W. \tag{3.3}$$

The family e is coherent in the sense that

$$\mathsf{ISTrees}_\gamma \circ \mathsf{e}_w \approx \mathsf{e}_{w'} \circ B_{\mathsf{node} \Rsh \gamma} \tag{3.4}$$

holds since $\mathsf{ISTrees}_\gamma$ and $B_{\mathsf{node} \Rsh \gamma}$ have the same function term. The family m is coherent in the sense of Definition 2.12 by Remark 3.9.

Remark 3.9. By (W2), the family of functions ist from (3.1) is coherent with respect to the setoid family $\mathsf{ISTrees}$ in the sense of Definition 2.12, *i.e.* for every $\gamma : w \approx_W w'$ there is

$$\mathsf{ist\text{-}coh}\, \gamma : \mathsf{ist}_w \approx \mathsf{ist}_{w'} \circ \mathsf{ISTrees}_\gamma.$$

The proof term $\mathsf{ist\text{-}coh}\, \gamma$ is obtained instantiating ϕ in Remark 3.2 with $b' \coloneqq \mathsf{ISTrees}_\gamma\, b$ and reflexivity on b'.

Similarly, the family of functions m from (3.3) is coherent with respect to the setoid family $\mathsf{ISTrees}$ in the sense of Definition 2.12, *i.e.* for every $\gamma : w \approx_W w'$ there is

$$\mathsf{m\text{-}coh}\, \gamma : \mathsf{m}_w \approx \mathsf{m}_{w'} \circ \mathsf{ISTrees}_\gamma. \tag{3.5}$$

The proof term $\mathsf{m\text{-}coh}\, \gamma$ is obtained from $\mathsf{ist\text{-}coh}$ using (3.4) and the fact that e_w is a (canonical regular) epi.

3.3. **Construction of the algebra map.** We construct the algebra map by showing that the function term sup lifts to an extensional function $\mathsf{s} : \mathsf{P}W \Rightarrow W$.

For $a : A$ and $f : B\, a \Rightarrow W$, let $f_0 \coloneqq \mathrm{pr}_1 f : |B|\, a \to \mathsf{W}_{|B|}$. The tree $w \coloneqq \mathsf{sup}\, a\, f_0$ is extensional by Lemma 3.5, so there is a function term

$$\mathsf{s} : \mathsf{P}W \to W. \tag{3.6}$$

Given explicitly by

$$\mathsf{s}\,(a, f) \coloneqq (\mathsf{sup}\, a\, f_0, \mathsf{dsup}\, (w, w)\, (\rho\, a)\, \varepsilon'_f) : W \tag{3.7}$$

where ρ is reflexivity of A, ε_f is obtained uncurrying $f\,\Rsh : \prod_{b,b'} b \approx_{B\,a} b' \to f\,b \approx_W f\,b'$ and $\mathsf{dsup}\,(w,w)\,(\rho\,a)\,\varepsilon_f : \mathsf{W}^{\mathsf{per}}_B\,w\,w$ is a proof that w is extensional.

Lemma 3.10. *The function term* $\mathsf{s} : \mathsf{P}W \to W$ *is extensional.*

Proof. By Definition 2.10, to have two equal elements in the domain is to have $a, a' : A$, $f : B\,a \Rightarrow W$, $f' : B\,a' \Rightarrow W$, and a term $\alpha : a \approx a'$such that $f \approx f' \circ B_\alpha$. Lemma 3.6 and Definition 3.1 yield the claim. □

We conclude this section with a proof that the algebra map $\mathsf{s} : \mathsf{P}W \Rightarrow W$ is invertible.

Proposition 3.11. *There is* $\mathsf{us} : W \Rightarrow \mathsf{P}W$ *such that* $\mathsf{s} \circ \mathsf{us} \approx id_W$ *and* $\mathsf{us} \circ \mathsf{s} \approx id_{\mathsf{P}W}$.

Proof. The function us maps an extensional tree $w \equiv \mathsf{sup}\,a\,f_0$ to the pair $(a, f) : \mathsf{P}W$, where $f : B\,a \Rightarrow W$ is the extensional function of w from Lemma 3.5. In formula

$$\mathsf{us}\,w \coloneqq (\mathsf{node}\,w, \mathsf{ist}_w) : \mathsf{P}W.$$

It is extensional because node is extensional, and because ist is coherent by Remark 3.9. The equation $\mathsf{us} \circ \mathsf{s} \approx \mathsf{id}_{\mathsf{P}W}$ is straightforward from the definitions. For $\mathsf{s} \circ \mathsf{us} \approx \mathsf{id}_W$ use Lemma 3.6. □

4. Initiality of the algebra of extensional trees

This section is mainly devoted to the characterisation of algebra morphisms as telescopic functions in Theorem 4.18. The existence of W-types in Std follows in Theorem 4.20.

4.1. Characterisation of algebra morphisms.

Throughout this section we fix setoids A and C, a setoid family $B : \mathsf{Fam}\,A$ and a P_B-algebra $a_C : \mathsf{P}_B C \Rightarrow C$. As in the previous section, we drop the index B from the polynomial functor P_B and the algebra of extensional trees W_B.

We begin with a simple observation. Recall from Definition 2.11.2 that $\mathsf{Alg}(\mathsf{s}, a_C)$ denotes the setoid of algebra morphisms from $\mathsf{s} : \mathsf{P}W \Rightarrow W$ to $a_C : \mathsf{P}C \Rightarrow C$, *i.e.* those functions $h : W \Rightarrow C$ such that

$$h \circ \mathsf{s} \approx a_C \circ \mathsf{P}h. \tag{4.1}$$

Every function $h : W \Rightarrow C$ gives rise to a family

$$F : \prod_{w:W} \mathsf{ISTrees}\,w \Rightarrow C \tag{4.2}$$

defined by $F_w \coloneqq h \circ \mathsf{m}_w$. It follows from (4.1) that the function h is an algebra morphism if and only if, for every $w \equiv \mathsf{s}(a, f)$,

$$h\,w \approx a_C(a, F_w \circ \mathsf{e}_w). \tag{4.3}$$

In particular, the value of the algebra morphism h at the tree w is obtained by "glueing" the values of h on the immediate subtrees of w, collected in F_w, according to the algebra a_C.

Note that the right-hand side of (4.3) makes sense for any family F as in (4.2). Let us say that a function $k : \mathsf{ISTrees}\,w \Rightarrow C$ is a *glueing datum in C for w*, and that a family F as in (4.2) is a *family of glueing data in C*. Two families F and F' of glueing data are equal if, for every $w : W$, the glueing data for w are equal, *i.e.* $F_w \approx F'_w$. Given a family F of glueing data in C as in (4.2), we can use (4.3) to define a function term $h : W \to C$. Proposition 4.3 isolates conditions on F for this term to be extensional and, further, an algebra morphism.

The problem of constructing an element of $\mathsf{Alg}(\mathsf{s}, a_C)$ could thus be reduced to the problem of finding a suitable family of glueing data in C and to prove that it is the unique such. We do so in Theorem 4.18, where we prove that $\mathsf{Alg}(\mathsf{s}, a_C)$ is in bijection with a subsetoid of families of glueing data in C that we name, in Definition 4.7, telescopic families. In Proposition 4.19 we construct a telescopic family by direct induction on the underlying W-type $\mathsf{W}_{|B|}$ of W. Its uniqueness is in Corollary 4.12. It follows that $\mathsf{Alg}(\mathsf{s}, a_C)$ is a singleton. This is in accordance with the result in [AGS17], that an algebra a is equivalent to a W-type if and only if the type of morphisms of algebras out of a is contractible, once we observe that, in the setoid interpretation, a type is contractible precisely when the corresponding setoid is a singleton.

There is no reason, at this stage, to prefer families of functions on immediate subtrees as in (4.2) over families of functions $B\,(\mathsf{node}\,w) \Rightarrow C$ on branches of trees, but our strategy does not seem to work with the latter. One motivation is discussed in Remark 4.6.

First, we give names to the two constructions outlined above.

Definition 4.1. Let $h : W \Rightarrow C$ be an extensional function. The *restriction of h to (immediate) subtrees* is the family of glueing data

$$\mathsf{restr}\,h \coloneqq \lambda w.h \circ \mathsf{m}_w : \prod_{w:W} \mathsf{ISTrees}\,w \Rightarrow C. \tag{4.4}$$

Let $F : \prod_{w:W} \mathsf{ISTrees}\,w \Rightarrow C$ be a family of glueing data. The *glueing of F (along a_C)* is the function term

$$\mathsf{glue}\,F \coloneqq \lambda w.a_C(\mathsf{node}\,w, F_w \circ \mathsf{e}_w) : W \to C. \tag{4.5}$$

We need one more definition to state the next result. Coherence for a family of extensional functions was defined in 2.12.

Definition 4.2. Let us denote by GlueD the setoid of families of glueing data in C. The underlying type is $\prod_{w:W} \mathsf{ISTrees}\,w \Rightarrow C$ and the equality between two families F and F' is

$$F \approx F' \;\coloneqq\; \prod_{w:W} F_w \approx_{(\mathsf{ISTrees}\,w \Rightarrow C)} F'_w.$$

The subsetoid of GlueD on the coherent families of glueing data is denoted $\mathsf{CohGlueD}$ and has underlying type

$$|\mathsf{CohGlueD}| \coloneqq \sum_{F:\mathsf{GlueD}} \mathsf{isCoh}_{\mathsf{ISTrees}}\,F.$$

Proposition 4.3.

(1) *For every function $h : W \Rightarrow C$, the family of glueing data* $\mathsf{restr}\,h$ *defined in* (4.4) *is coherent.*
(2) *For every coherent family of glueing data F, the function* $\mathsf{glue}\,F$ *defined in* (4.5) *is extensional.*
(3) *The function* $\mathsf{restr} : (W \Rightarrow C) \to \mathsf{CohGlueD}$ *is extensional.*
(4) *The function* $\mathsf{glue} : \mathsf{CohGlueD} \to (W \Rightarrow C)$ *is extensional.*
(5) *For every $h : W \Rightarrow C$*

$$\mathsf{glue}\,(\mathsf{restr}\,h) \approx h \;\longleftrightarrow\; h \circ \mathsf{s} \approx a_C \circ \mathsf{P}h.$$

(6) *For every coherent family $F : \prod_{w:W} \mathsf{ISTrees}\,w \Rightarrow C$ of glueing data*

$$\mathsf{restr}\,(\mathsf{glue}\,F) \approx F \;\longrightarrow\; \mathsf{glue}\,F \circ \mathsf{s} \approx a_C \circ \mathsf{P}(\mathsf{glue}\,F).$$

Proof. (1) For every $\gamma : w \approx_W w'$, it is $h \circ \mathsf{m}_w \approx h \circ \mathsf{m}_{w'} \circ \mathsf{ISTrees}_\gamma$ since m_w is coherent by (3.5) in Remark 3.9, and h is extensional.

(2) This is Lemma 2.13 together with extensionality of a_C.

(3) If $h \approx h'$, then for every $w : W$ it is $h \circ \mathsf{m}_w \approx h' \circ \mathsf{m}_w$ by extensionality of h.

(4) Let $\varphi : F \approx F'$. If $F_w \approx F'_w$, then by extensionality of a_C and proof irrelevance of B from (2.6) it is

$$a_C(a, F_w) \approx_C a_C(a, F'_w).$$

(5) The claim follows from the fact that, for every $w \equiv \mathsf{s}(a, f)$, it is

$$\begin{aligned} \mathsf{glue}\,(\mathsf{restr}\,h)\,w &\equiv a_C(a, h \circ \mathsf{m}_w \circ \mathsf{e}_w) \\ &\approx a_C(a, h \circ f) \\ &\approx a_C \circ \mathsf{P}h(a, f) \\ &\approx a_C \circ \mathsf{P}h \circ \mathsf{us}\,w \end{aligned} \tag{4.6}$$

by, in order, (4.4) and (4.5), Remark 3.8, definition of polynomial functor in 2.10 and definition of us in Proposition 3.11.

(6) Unfolding the right hand side for $w \equiv \mathsf{s}(a, f)$ yields

$$a_C(a, F_w \circ \mathsf{e}_w) \approx_C a_C(a, \lambda b.a_C(\mathsf{node}\,(f\,b), F_{(f\,b)} \circ \mathsf{e}_{(f\,b)}))$$

By extensionality of a_C and proof irrelevance (2.6), this holds if for every $b : B\,a$ it is

$$F_w\,b \approx_C a_C(\mathsf{node}\,(f\,b), F_{(f\,b)} \circ \mathsf{e}_{(f\,b)}).$$

Now, by definitions (4.4) and (4.5) it is

$$\mathsf{restr}\,(\mathsf{glue}\,F)\,w\,b \equiv a_C(\mathsf{node}\,(f\,b), F_{(f\,b)} \circ \mathsf{e}_{(f\,b)})$$

and the claim follows. □

Corollary 4.4. *The functions* restr *and* glue *establish a bijection between the setoid* $\mathsf{Alg}(\mathsf{s}, a_C)$ *and those coherent families* F *of glueing data in* C *such that* $\mathsf{restr}\,(\mathsf{glue}\,F) \approx F$.

Proof. Proposition 4.3.5 entails that $\mathsf{restr}\,(\mathsf{glue}\,(\mathsf{restr}\,h)) \approx \mathsf{restr}\,h$ for every algebra morphism h. Conversely, If F is such that $\mathsf{restr}\,(\mathsf{glue}\,F) \approx F$, then $\mathsf{glue}\,F$ is in $\mathsf{Alg}(\mathsf{s}, a_C)$ by Proposition 4.3.6. □

The bijection in Corollary 4.4 suggests a recursive characterisation of algebra morphisms into C as those morphisms obtained by glueing a coherent family of glueing data in C, such that each component F_w is itself obtained by glueing a suitable family, and so on. In order to make this intuition precise, we need to bring coherent families of glueing data and glueing one "subtree level" up. We do not need to go any higher thanks to dependent W-types, which will cover the other cases for us in Definition 4.7, see also Remark 4.9. The characterisation is accomplished in Theorem 4.18.

For w an extensional tree, we say that a family

$$G : \prod_{s : \mathsf{ISTrees}\,w} \mathsf{ISTrees}\,(\mathsf{m}_w\,s) \Rightarrow C.$$

is a *family of glueing subdata for* w *in* C. We write G_s for $G\,s$ as we do for families of glueing data on subtrees, and we refer to terms of type $\mathsf{ISTrees}\,(\mathsf{m}_w\,s)$ as *2-subtrees of* w.

By extensionality of m_w, the family of setoids $\lambda b.\mathsf{ISTrees}\,(\mathsf{m}_w\,b)$ over $\mathsf{ISTrees}\,w$ is a setoid family in the same way as for the family of subtrees $\mathsf{ISTrees}$ in Definition 3.7. We can thus

form the setoid $\mathsf{CohGlueSubD}\,w$ of *coherent families of glueing subdata for* w, as we did in Definition 4.2 for coherent families of glueing data of subtrees.

In fact, $\mathsf{CohGlueSubD}$ is another setoid family over W, whose transport function

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

for $\gamma : w \approx_W w'$ is defined on G and $s : \mathsf{ISTrees}\,w'$ as

$$\mathsf{CohGlueSubD}_\gamma\,G\,s \coloneqq G_{(\mathsf{ISTrees}_{\gamma^{-1}}\,s)} \circ \mathsf{ISTrees}_{(\mathsf{m\text{-}coh}\,\gamma^{-1}\,s')} \tag{4.7}$$

where $\mathsf{m\text{-}coh}\,\gamma^{-1}\,s' : \mathsf{m}\,s' \approx \mathsf{m}\,(\mathsf{ISTrees}_{\gamma^{-1}}\,s')$. It follows that

$$G \approx_\gamma G' \;\longleftrightarrow\; \prod_{s:\mathsf{ISTrees}\,w} G_s \approx G'_{(\mathsf{ISTrees}_\gamma\,s)} \circ \mathsf{ISTrees}_{(\mathsf{m\text{-}coh}\,\gamma\,s)} \tag{4.8}$$

for all $G : \mathsf{CohGlueSubD}\,w$ and $G' : \mathsf{CohGlueSubD}\,w'$.

As in (4.5) for every family G of glueing subdata for w there is a function term

$$\mathsf{subglue}_w\,G : \mathsf{ISTrees}\,w \to C$$

defined on $s : \mathsf{ISTrees}\,w$ by

$$\mathsf{subglue}_w\,G\,s \coloneqq a_C\,\left(\mathsf{node}\,(\mathsf{m}_w\,s), G_s \circ \mathsf{e}_{(\mathsf{m}_w\,s)}\right). \tag{4.9}$$

Lemma 4.5.

(1) *For every coherent family of glueing subdata* $G : \mathsf{CohGlueSubD}\,w$*, the function* $\mathsf{subglue}_w\,G$ *defined in* (4.9) *is extensional.*

(2) *For every* $w : W$*, the function* $\mathsf{subglue}_w : \mathsf{CohGlueSubD}\,w \to (\mathsf{ISTrees}\,w \Rightarrow C)$ *is extensional.*

(3) *For every* $\gamma : w \approx_W w'$*,* $G : \mathsf{CohGlueSubD}\,w$ *and* $G' : \mathsf{CohGlueSubD}\,w'$ *there is*

$$\mathsf{subglue\text{-}coh}\,\gamma : G \approx_\gamma G' \longrightarrow \mathsf{subglue}_w\,G \approx \mathsf{subglue}_{w'}\,G' \circ \mathsf{ISTrees}_\gamma.$$

Proof. (1) It follows from Lemma 2.13 and extensionality of a_C.

(2) It follows from (3) with reflexivity as γ.

(3) Let γ, G and G' be as above, and let $\varphi : G \approx_\gamma G'$ and $s : \mathsf{ISTrees}\,w$. Using extensionality of a_C, it is enough to show

$$(\mathsf{node}\,(\mathsf{m}_w\,s), G_s) \approx \left(\mathsf{node}\,(\mathsf{m}_{w'}\,(\mathsf{ISTrees}_\gamma\,s)), G'_{(\mathsf{ISTrees}_\gamma\,s)}\right).$$

For the first component we may take $\alpha \coloneqq \mathsf{node}\,\Rsh(\mathsf{m\text{-}coh}\,\gamma)$ from Remark 3.9. It remains to show that, for every $s : \mathsf{ISTrees}\,w$,

$$G_s \approx G'_{(\mathsf{ISTrees}_\gamma\,s)} \circ \mathsf{ISTrees}_{(\mathsf{m\text{-}coh}\,\gamma)}.$$

By (4.8) it is enough to use φ. □

Remark 4.6. Lemma 4.5 is the reason for choosing to work with the setoid family of immediate subtrees instead of the setoid family of branches of trees from Section 3.2.

Indeed, Proposition 4.3 works equally well with immediate subtrees replaced by branches of trees, in the sense that there are function terms between extensional functions $W \Rightarrow C$ and families of glueing data on branches satisfying the same properties, mutatis mutandis, of restr and glue. Problems arise at the next subtree level.

If we say that, for an extensional tree w, a term of type

$$\prod_{b:B\,(\mathsf{node}\,w)} B\,(\mathsf{node}\,(\mathsf{ist}_w\,b)) \Rightarrow C$$

is a *family of glueing subdata on branches for w in C*, we can then form, for every $w : W$, the setoid CB_w of coherent families of glueing subdata on branches for w. Nevertheless, it seems that transport of families of glueing subdata on branches over $\gamma : w \approx_W w'$ does not preserve coherence. It follows that, contrary to $\mathsf{CohGlueSubD}$, the family $\lambda w.CB_w$ is not a setoid family on W. In particular, there is no function term like $\mathsf{CohGlueSubD}_\gamma$ from (4.7), which is needed, together with Lemma 4.5.3, in a crucial step in the proof of Lemma 4.13. The issue seems to arise because branches, contrary to subtrees, do not form a subsetoid of W.

Note that there is no version of restr for 2-subtrees: a function $k : \mathsf{ISTrees}\, w \Rightarrow C$ cannot be restricted to $\mathsf{ISTrees}\,(\mathsf{m}_w\, s)$ for some $s : \mathsf{ISTrees}\, w$ simply because an immediate subtree of $\mathsf{m}_w\, s$ is not an immediate subtree of w, that is, being an immediate subtree is not a transitive property. This entails that, in order to say that k is a glueing, we need to assume that a coherent family of glueing subdata for w exists.

Definition 4.7. Let $w \equiv \mathsf{s}(a, f)$ be an extensional tree. A function $k : \mathsf{ISTrees}\, w \Rightarrow C$ is *telescopic over w* if

(1) the function k is the glueing of a coherent family of glueing subdata $G : \mathsf{CohGlueSubD}\, w$, *i.e.* $k \approx \mathsf{subglue}_w\, G$, and
(2) for every $s : \mathsf{ISTrees}\, w$, the function $G_s : \mathsf{ISTrees}\,(f\, s) \Rightarrow C$ is telescopic over $f\, s$.

Accordingly, we define the type family

$$\mathsf{isTelescopic} : \left(\sum_{w:W} \mathsf{ISTrees}\, w \Rightarrow C\right) \to \mathsf{U}$$

of proofs that a function on immediate subtrees is telescopic as the dependent W-type on $I \coloneqq \sum_{w:W}(\mathsf{ISTrees}\, w \Rightarrow C)$ with family of nodes

$$N \coloneqq \lambda(w, k). \sum_{G:\mathsf{CohGlueSubD}\, w} k \approx \mathsf{subglue}_w\, G : \left(\sum_{w:W} \mathsf{ISTrees}\, w \Rightarrow C\right) \to \mathsf{U}$$

branching family

$$Br \coloneqq \lambda(w, k), (G, \_).\mathsf{ISTrees}\, w : \prod_{(w,k):I} N\,(w, k) \to \mathsf{U}$$

and arity function

$$ar \coloneqq \lambda(w, k), (G, \_), s.(\mathsf{m}_w\, s, G_s) : \prod_I \prod_N \mathsf{ISTrees}\, w \to \left(\sum_{w:W} \mathsf{ISTrees}\, w \Rightarrow C\right).$$

Henceforth we write $\mathsf{isTelescopic}_w\, k$ for $\mathsf{isTelescopic}\,(w, k)$ and we may drop the subscript w if it is clear from context.

Definition 4.8. A family F of glueing data in C is *telescopic* if each of its components is a telescopic function over w, that is, if

$$\prod_{w:W} \mathsf{isTelescopic}\, F_w$$

is inhabited. We say 'telescopic family' to mean 'telescopic family of glueing data in C'.

Telescopic families of glueing data form a subsetoid $\mathsf{TelGlueD}$ of GlueD.

Remark 4.9. If we say, similarly, that a family G of glueing subdata for w is telescopic over w if each component G_s is telescopic over the immediate subtree $\mathsf{m}_w\, s$, then Definition 4.7 can be phrased as: a function is telescopic over w if it is the glueing of a telescopic family of glueing subdata for w.

The canonical components of a proof that a function $k : \mathsf{ISTrees}\, w \Rightarrow C$ is telescopic are as follows, see also Figure 2. From the dependent node function (2.1) we obtain, for every $T : \mathsf{isTelescopic}_w\, k$, a coherent family of glueing subdata

$$\mathsf{csfam}\, k\, T : \mathsf{CohGlueSubD}\, w \tag{4.10}$$

and a proof

$$\mathsf{isglue}\, k\, T : k \approx \mathsf{subglue}_w\, (\mathsf{csfam}\, k\, T), \tag{4.11}$$

that k is a glueing of the family $\mathsf{csfam}\, k\, T$. The dependent subtree function (2.2) yields a proof

$$\mathsf{telsfam}\, k\, T : \prod_{(s:\mathsf{ISTrees}\, w)} \mathsf{isTelescopic}_{(\mathsf{m}_w\, s)}\, (\mathsf{csfam}\, k\, T\, s). \tag{4.12}$$

that every function in the family $\mathsf{csfam}\, k\, T$ is telescopic, that is, the family csfam of glueing subdata is itself telescopic, see Remark 4.9.

Clearly, terms in (4.10) and (4.11) are proof terms for 4.7.1, and the term in (4.12) is a proof term for 4.7.2.

As we will extensively use the elimination principle in Figure 2 of the dependent W-type of telescopic functions from Definition 4.7, let us unfold its inductive hypothesis too. Let $V : \prod_{w,k} \mathsf{isTelescopic}\, k \to \mathsf{U}$ be a (small) type family. The inductive hypothesis tells us that, for $T \equiv \mathsf{dsup}\, (w, k)\, G\, E : \mathsf{isTelescopic}\, k$ and for every $s : \mathsf{ISTrees}\, w$, there is a term

$$IH\, s : V\, (\mathsf{m}_w\, s)\, G_s\, (E\, s) \tag{4.13}$$

where $G \equiv \mathsf{csfam}\, T$ is the coherent family of glueing subdata obtained from T as in (4.10), and the term $E \equiv \mathsf{telsfam}\, T$ is the proof (4.12) that every G_s is telescopic over $\mathsf{m}_w\, s$.

The fact that functions $\mathsf{ISTrees}\, w \Rightarrow C$ cannot be restricted to families of glueing subdata, discussed right before Definition 4.7, does not prevent us from showing that restrictions of algebra morphisms are telescopic, since a function $W \Rightarrow C$ can be restricted to any "subtree level". In particular, to immediate subtrees and 2-subtrees.

Lemma 4.10. *Let $h : W \Rightarrow C$ be an algebra morphism. Then for every $w : W$, the function*

$$\mathsf{restr}\, h\, w := h \circ \mathsf{m}_w : \mathsf{ISTrees}\, w \Rightarrow C$$

defined in (4.4) *is telescopic over w.*

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

$$\mathsf{W}^{\mathsf{per}}_B\, w\, w \quad \longrightarrow \mathsf{isTelescopic}\, (h \circ \mathsf{m}_w).$$

Note that, whenever we wish to apply the inductive hypothesis to a subtree index of w, we have to provide a proof that the subtree is extensional. Such a proof will always be available since w is extensional by assumption and immediate subtrees of extensional trees are extensional by Lemma 3.5. So we may assume without loss of generality that all the trees that we will be dealing with are extensional.

Let then $w \equiv \mathsf{sup}\, a\, f$. By Definition 4.7, in order to apply dsup, we first need to provide a coherent family of glueing subdata $G : \mathsf{CohGlueSubD}\, w$ and a proof that $h \circ \mathsf{m}_w \approx \mathsf{subglue}_w\, G$.

The family G of glueing subdata consists of the restrictions of h to 2-subtrees of w, namely

$$G_s \coloneqq h \circ \mathsf{m}_{(\mathsf{m}\, s)} : \mathsf{ISTrees}\,(\mathsf{m}_w\, s) \Rightarrow C$$

for every $s : \mathsf{ISTrees}\, w$. Its coherence follows as in 4.3.1 from coherence of m and extensionality of h. For every $s : \mathsf{ISTrees}\, w$ it is

$$\begin{aligned} h\,(\mathsf{m}_w\, s) &\approx a_C \circ (\mathsf{P}h) \circ \mathsf{us}\,(\mathsf{m}_w\, s) \\ &\approx a_C\,(\mathsf{node}\,(\mathsf{m}_w\, s), h \circ \mathsf{m}_{(\mathsf{m}\, s)}) \\ &\approx \mathsf{subglue}_w\, G\, s \end{aligned}$$

by the fact that h is an algebra morphism, definition of us and P, and definition of $\mathsf{subglue}$ in 4.9.

The inductive hypothesis witnesses that each restriction $h \circ \mathsf{m}_{(\mathsf{m}\, s)}$ is telescopic. □

It follows from this lemma that the function $\mathsf{restr} : (W \Rightarrow C) \Rightarrow \mathsf{GlueD}$ from Proposition 4.3.3 restricts to a function

$$\mathsf{restr} : \mathsf{Alg}(\mathsf{s}, a_C) \Rightarrow \mathsf{TelGlueD} \tag{4.14}$$

that maps an algebra morphism to its family of restrictions as defined in (4.4). Because of the bijection in Corollary 4.4, to see this we could have equally well proved that every coherent family F of glueing data such that $\mathsf{restr}\,(\mathsf{glue}\, F) \approx F$ is a telescopic family. Our choice is justified by the fact that functions and their restrictions are simpler objects than families of glueing data and their restrictions. The fact that every such F is telescopic is an immediate consequence of the Lemma above and Proposition 4.3.6.

Our aim is now to prove that glue lifts to an inverse of restr. To do so we need to prove that a telescopic family is coherent. Then by Proposition 4.3.6 it will be enough to show that a telescopic family is the restriction of its own glueing.

First we need some technical lemmas about telescopic functions. We begin by showing that any two functions which are telescopic over the same w are necessarily equal. A more general version is in Lemma 4.14.

Lemma 4.11. *Let $w : W$ and $k, k' : \mathsf{ISTrees}\, w \Rightarrow C$. Then*

$$\mathsf{isTelescopic}\, k \to \mathsf{isTelescopic}\, k' \to k \approx k'.$$

Proof. This is proven by induction on $T \equiv \mathsf{dsup}\,(w, k)\, G\, E : \mathsf{isTelescopic}\, k$ into the type

$$\prod_{k'} \mathsf{isTelescopic}\, k' \to k \approx k'.$$

Let $G' \coloneqq \mathsf{csfam}\, w\, k'\, T'$ be the coherent family of glueing subdata given by the assumption T' that k' is telescopic, as in (4.10). Since k and k' are the glueing of G and G', respectively, it is enough to show that

$$\mathsf{subglue}_w\, G \approx \mathsf{subglue}_w\, G'.$$

Using extensionality of $\mathsf{subglue}_w$ from Lemma 4.5.2 this reduces to show that the two families G and G' of glueing subdata are equal, namely that for every $s : \mathsf{ISTrees}\, w$

$$G_s \approx G'_s.$$

This is the inductive hypothesis (4.13) applied to the right-hand function and the proof from (4.12) that it is telescopic. □

Corollary 4.12. *The setoid* TelGlueD *is a subsingleton, that is,* $F \approx F'$ *for any two telescopic families* F *and* F'*.*

Proof. Straightforward from Definition 4.8 and Lemma 4.11. □

The next lemma shows that telescopic functions are stable under transport over W.

Lemma 4.13. *Let* $\gamma : w \approx_W w'$ *and* $k : \mathsf{ISTrees}\, w \Rightarrow C$*. Then*

$$\mathsf{isTelescopic}_w\, k \to \mathsf{isTelescopic}_{w'}\, (k \circ \mathsf{ISTrees}_{\gamma^{-1}}).$$

Proof. This is proven by induction on $T \equiv \mathsf{dsup}\, (w, k)\, G\, E : \mathsf{isTelescopic}\, k$ into the type

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

and we work towards applying dsup, that is conditions (1) and (2) in Definition 4.7.

Since k is obtained applying $\mathsf{subglue}_w$ to G by (4.11), it is

$$k \circ \mathsf{ISTrees}_{\gamma^{-1}} \approx \mathsf{subglue}_{w'}\, (\mathsf{CohGlueSubD}_\gamma\, G)$$

by Lemma 4.5.3. It remains to provide a branching function to establish condition 4.7.2. This amounts to show that, for each $s' : \mathsf{ISTrees}\, w'$, the function

$$\mathsf{CohGlueSubD}_\gamma\, G_{s'} \equiv G_s \circ \mathsf{ISTrees}_{(\mathsf{m\text{-}coh}\, \gamma^{-1}\, s')} : \mathsf{ISTrees}\, (\mathsf{m}_{w'}\, s') \Rightarrow C$$

from (4.7) is telescopic, where $s \coloneqq \mathsf{ISTrees}_{\gamma^{-1}}\, s' : \mathsf{ISTrees}\, w$. This is the inductive hypothesis in (4.13) applied to

$$(\mathsf{m\text{-}coh}\, \gamma^{-1}\, s')^{-1} : \mathsf{m}_w\, s \approx \mathsf{m}_{w'}\, s'$$

from Remark 3.9. □

Lemma 4.14. *Let* $\gamma : w \approx_W w'$*,* $k : \mathsf{ISTrees}\, w \Rightarrow C$ *and* $k' : \mathsf{ISTrees}\, w' \Rightarrow C$*. Then*

$$\mathsf{isTelescopic}_w\, k \to \mathsf{isTelescopic}_{w'}\, k' \to k \approx k' \circ \mathsf{ISTrees}_\gamma.$$

Proof. Straightforward from Lemmas 4.11 and 4.13. □

Recall that a family of glueing data is telescopic if each of its components is a telescopic function. We see in particular that:

Corollary 4.15. *Telescopic families are coherent, that is, the setoid* TelGlueD *is a subsetoid of* CohGlueD*.*

In the following two results we make sure that glue maps telescopic families to algebra morphisms.

Lemma 4.16. *Let* F *be a telescopic family. Then the family* $\mathsf{restr}\, (\mathsf{glue}\, F)$ *of glueing data is also telescopic.*

Proof. We need to show that, for every $w : W$, the function

$$\mathsf{restr}\, (\mathsf{glue}\, F)\, w \equiv (\mathsf{glue}\, F) \circ \mathsf{m}_w : \mathsf{ISTrees}\, w \Rightarrow C$$

is telescopic, and we do so by proving that it satisfies conditions (1) and (2) in Definition 4.7.

As family of glueing subdata for w we can take the family G defined by $G_s \coloneqq F_{(\mathsf{m}_w\, s)}$. Its coherence follows as for Corollary 4.15 from Lemma 4.14 and the fact that F is telescopic. For $s : \mathsf{ISTrees}\, w$ it is

$$\begin{aligned}\mathsf{restr}\,(\mathsf{glue}\, F)\, w\, s &\equiv (\mathsf{glue}\, F)\,(\mathsf{m}_w\, s)\\ &\equiv a_C(\mathsf{node}\,(\mathsf{m}_w\, s), F_{(\mathsf{m}_w\, s)})\\ &\equiv \mathsf{subglue}_w\, G\end{aligned}$$

by definition of glue in (4.5) and definition of $\mathsf{subglue}$ in (4.9). Since for every w, the function F_w is telescopic, this is the case in particular for $G_s \equiv F_{\mathsf{m}_w\, s}$ for every $s : \mathsf{ISTrees}\, w$. □

Corollary 4.17. *For every telescopic family F, it is*

$$\mathsf{restr}\,(\mathsf{glue}\, F) \approx F$$

and the function $\mathsf{glue}\, F$ *is in* $\mathsf{Alg}(\mathsf{s}, a_C)$*, that is*

$$\mathsf{glue}\, F \circ \mathsf{s} \approx a_C \circ \mathsf{P}(\mathsf{glue}\, F).$$

Proof. The first equality follows from Lemma 4.16 and Lemma 4.11. So the function $\mathsf{glue}\, F$ is an algebra morphism by Proposition 4.3.6. □

It follows that the function $\mathsf{glue} : \mathsf{CohGlueD} \Rightarrow (W \Rightarrow C)$ defined in (4.5) restricts to a function

$$\mathsf{glue} : \mathsf{TelGlueD} \Rightarrow \mathsf{Alg}(\mathsf{s}, a_C) \tag{4.15}$$

from telescopic families to algebra morphisms.

Theorem 4.18. *The functions* restr *from* (4.14) *and* glue *from* (4.15)

$$\mathsf{Alg}(\mathsf{s}, a_C) \underset{\mathsf{glue}}{\overset{\mathsf{restr}}{\rightleftarrows}} \mathsf{TelGlueD}$$

establish a bijection between the setoid of algebra morphisms from s *to* a_C *and the setoid of telescopic families of glueing data in C.*

Proof. Proposition 4.3.5 yields that $\mathsf{glue}\,(\mathsf{restr}\, h) \approx h$ for every $h : \mathsf{Alg}(\mathsf{s}, a_C)$. The other equality is in Corollary 4.17. □

As an immediate consequence, we see from Corollary 4.12 that the setoid of algebra morphisms is a subsingleton. It is then clear that $\mathsf{Alg}(\mathsf{s}, \mathsf{s})$ is a singleton, and that the only telescopic family in this case is m. It only remains to construct an inhabitant in the general case.

Proposition 4.19. *There is a telescopic family of glueing data in C, i.e. a term*

$$\mathsf{telfam} : \mathsf{TelGlueD}.$$

Proof. Given an extensional tree $w : W$, we need to show that there is a telescopic function

$$\mathsf{telfam}_w : \mathsf{ISTrees}\, w \Rightarrow C.$$

The proof is by W-elimination on $w : \mathsf{W}_{|B|}$ into the type

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

As in the proof of Lemma 4.10, we may assume without loss of generality that all the trees in the proof are extensional.

We have a tree of the form $w \equiv \mathsf{sup}\, a\, f$ and the inductive hypothesis consists, for every $s : \mathsf{ISTrees}\, w$, of a telescopic function

$$G_s : \mathsf{ISTrees}\, (\mathsf{m}_w\, s) \Rightarrow C.$$

This is the same as a telescopic family G of glueing subdata for w, see Remark 4.9.

It follows from Lemma 4.14 that G is coherent, so we may define

$$\mathsf{telfam}_w \coloneqq \mathsf{subglue}_w\, G : \mathsf{ISTrees}\, w \Rightarrow C$$

as the glueing of a telescopic family G of glueing subdata for w. It is a telescopic function by definition. □

4.2. **Initiality.** We have finally reached our main result.

Theorem 4.20. *For every setoid family B over a setoid A, the associated polynomial endofunctor P has an initial algebra.*

It follows that the category of setoids Std has initial algebras for polynomial endofunctors.

Proof. Given a setoid family B over A, the initial algebra for the polynomial functor P is the algebra $\mathsf{s}_B : \mathsf{P}W \Rightarrow W$ of extensional trees constructed in Section 3.1. It follows by Theorem 4.18 that, for every algebra a_C, the only morphism of algebras from s to a_C is the function $\mathsf{glue}(\mathsf{telfam}) : W \Rightarrow C$, where $\mathsf{glue} : \mathsf{TelGlueD} \Rightarrow \mathsf{Alg}(\mathsf{s}_B, a_C)$ is the glueing function from Theorem 4.18 and telfam is the only telescopic family of glueing data in C, that we constructed in Proposition 4.19 and proved unique in Corollary 4.12.

It is straightforward to verify that every polynomial functor P on Std has initial algebra $\mathsf{s}_B \circ i_W : PW \Rightarrow W$, where i is a natural isomorphism from P to P as in Definition 2.10. □

Gambino and Hyland have proved that, in a locally cartesian closed category, dependent W-types can be constructed from non-dependent ones, thus providing justification for an analogous result in extensional type theory claimed in [PS89]. As a consequence of Theorem 4.20, we expect that setoids have dependent W-types too, *i.e.* that the category of Std has initial algebras for *dependent* polynomial endofunctors.

5. Trees on discrete setoids

In this section we work with the additional assumption that our theory has identity types [NPS90, Chapter 20.4], see also [UFP13, Chapter 1.12]. We unfold the definition of the setoid of extensional trees in the case of a *discrete* setoid family, *i.e.* a setoid family coming from an extensional function between discrete setoids. We then compare it to the discrete setoid on the W-type of the underlying type family. The main result is Theorem 5.8.

In comparing the two, we will also make use of function extensionality. This is because the functions of immediate subtrees of two matching trees are only required to be pointwise equal, as in the right-hand side of (W2). Therefore, rather than to compare the two directly, we prefer to identify another (total) equivalence relation on $\mathsf{W}_{|B|}$, which is logically equivalent to the identity type on $\mathsf{W}_{|B|}$ in the presence of function extensionality, and to compare it to $\mathsf{W}_B^{\mathsf{per}}$ in the general case.

Recall that $x =_X x'$ denotes the identity type of two terms $x, x' : X$. For a type family $Y : X \to \mathsf{U}$, we denote as $Y_\xi : Y\, x \to Y\, x'$ the transport function term given by identity elimination on $\xi : x =_X x'$, and we write $y =_\xi y'$ for the identity type $Y_\xi\, y =_{(Y\, x')} y'$.

5.1. **Discrete setoid families.** We begin by identifying those setoid families that correspond, under Theorem 2.8, to functions between discrete setoids.

Definition 5.1. A setoid family $B : \mathsf{Fam}\, X$ over the discrete setoid on X is *discrete over* X if, for every $x : X$ and $b, b' : B\, x$,

$$b \approx_{B\, x} b' \longleftrightarrow \sum_{l : x =_X x} |B|_l\, b =_{B\, x} b'$$

where $|B|_\xi : B\, x \to B\, x$ is the transport function of the underlying type family $|B|$ of B given by elimination of the identity type.

The equivalence in Theorem 2.8 restricts to an equivalence between the full subcategory of Std/X on the functions into X from a discrete setoid, and the full subcategory of $\mathsf{Fam}\, X$ on the discrete setoid families over X.

It follows in particular that every type family $Y : X \to \mathsf{U}$ gives rise to a discrete setoid family over X, by equipping each fibre $Y\, x$ with the equality

$$y \approx y' \coloneqq \sum_{\xi : x =_X x} Y_\xi\, y = y'$$

where $Y_\xi : Y\, x \to Y\, x$ is the transport function given by elimination of the identity type.

5.2. **Pointwise equality of trees.** Next, we look for an equivalent characterisation of the discrete setoid on a W-type in the presence of function extensionality.

Definition 5.2. Let $Y : X \to \mathsf{U}$ be a family of small types. Two trees $w \equiv \mathsf{sup}\, x\, f$ and $w' \equiv \mathsf{sup}\, x'\, f'$ in W_Y are *pointwise equal* if $\xi : x =_X x'$ and, for every $y \in Y\, x$ and $y' : Y\, x'$, if $y =_\xi y'$ then the immediate subtrees $f\, y$ and $f'\, y'$ are pointwise equal.

The type family $\mathsf{ptEq}_Y : \mathsf{W}_Y \to \mathsf{W}_Y \to \mathsf{U}$ of proofs that w and w' are pointwise is defined as the (curried version of the) dependent W-type on $I \coloneqq \mathsf{W}_Y \times \mathsf{W}_Y$ with family of nodes

$$N \coloneqq \lambda(w, w').\mathsf{node}\, w =_X \mathsf{node}\, w' : \mathsf{W}_Y \times \mathsf{W}_Y \to \mathsf{U}$$

branching family

$$Br \coloneqq \lambda(w, w'), p. \sum_{y, y'} y =_\xi y' : \prod_I N\, (w, w') \to \mathsf{U}$$

and arity function

$$ar := \lambda(w, w'), p, (y, y', q).(\mathsf{ist}_w\, y, \mathsf{ist}_{w'}\, y') : \prod_I \prod_N Br((w, w'), p) \to \mathsf{U}.$$

It is possible to show that ptEq_Y is a symmetric and transitive relation on W_Y as for $\mathsf{W}_B^{\mathsf{per}}$ in Proposition 3.3. Reflexivity follows from the following lemma.

Lemma 5.3. *For every* $w, w' : \mathsf{W}_Y$

$$w =_{\mathsf{W}_Y} w' \longrightarrow \mathsf{ptEq}_Y\, w\, w'.$$

Thus we have a setoid $\widetilde{\mathsf{W}}_Y$ of pointwise-equal trees, that is the type W_Y with ptEq as equality, together with an extensional function

$$q_Y : \mathsf{W}_Y \Rightarrow \widetilde{\mathsf{W}}_Y \tag{5.1}$$

from the discrete setoid on W_Y which is in fact a (canonical) quotient map in Std, cf. Remark 3.8.

Proof of Lemma 5.3. The proof is by induction on $w \equiv \mathsf{sup}\, x\, f$ and the inductive hypothesis tells us that if a tree is equal to an immediate subtree of w, then it is pointwise equal to it.

By induction on $r : w = w'$, it is enough to show that $\mathsf{ptEq}_Y\, w\, w$. A canonical term is given taking the canonical proof of $x =_X x$ and proving that, for every $y, y' : Y\, x$, if $y =_{(Y\, x)} y'$ then the trees $f\, y$ and $f\, y'$ are pointwise equal. This follows from the inductive hypothesis using the fact that every function term—f in particular, is extensional with respect to the identity type. □

Unsurprisingly, it is for the converse that we need function extensionality.

Lemma 5.4. *Assume that the theory has function extensionality, i.e. that there is a constant* funext *as given in the rule below.*

$$\frac{X, Y : \mathsf{U} \qquad f, g : X \to Y \qquad H : \prod_{x:X} f\, x =_Y g\, x}{\mathsf{funext}\, H : f =_{(X \to Y)} g} \tag{5.2}$$

Then for every $w, w' : \mathsf{W}_Y$

$$\mathsf{ptEq}_Y\, w\, w' \longrightarrow w =_{\mathsf{W}_Y} w'.$$

Proof. Let $w \equiv \mathsf{sup}\, x\, f$ and $w' \equiv \mathsf{sup}\, x'\, f$'. The proof is by induction on the term $E : \mathsf{ptEq}\, w\, w'$. To conclude $w = w'$ it is enough to show that there is $\xi : x =_X x'$ such that $f = f' \circ Y_\xi$. The existence of p follows by definition of $\mathsf{ptEq}\, w\, w'$. By function extensionality and the inductive hypothesis it is then enough to show that, for every $y : Y\, x$, the immediate subtrees $f\, y$ and $f'\, (Y_\xi\, y)$ are pointwise equal. This also follows by definition of $\mathsf{ptEq}\, w\, w'$. □

It follows that, in the presence of function extensionality, the discrete setoid on W_Y is isomorphic to the setoid $\widetilde{\mathsf{W}}_Y$ of pointwise-equal trees via the function q_Y in (5.1).

5.3. **Equivariant trees.** Now we look for a characterisation of the setoid W_B of extensional trees of a discrete setoid family B as subsetoid of $\widetilde{W}_{|B|}$, *i.e.* as a setoid of pointwise equal trees.

Definition 5.5. Let $Y : X \to \mathsf{U}$ be a type family. A tree $w \equiv \mathsf{sup}\, x\, f : \mathsf{W}_Y$ is *equivariant* if, for every $l : x =_X x$ and every $y : Y\, x$, the immediate subtrees $f\, (Y_l\, y)$ and $f\, y$ are pointwise equal, and every immediate subtree of w is equivariant.

The type family $\mathsf{isEquivariant} : \mathsf{W}_Y \to \mathsf{U}$ of proofs that a tree is equivariant is defined as the dependent W-type in $I \coloneqq \mathsf{W}_Y$ with families of nodes and branches

$$N \coloneqq \lambda w. \prod_{(l:\mathsf{node}\, w =_X \mathsf{node}\, w)} \prod_{(y:Y\, (\mathsf{node}\, w))} \mathsf{ptEq}_Y(f\, (Y_l\, y), f\, y), \qquad Br \coloneqq \lambda w, \_.Y\, (\mathsf{node}\, w)$$

and arity function

$$ar \coloneqq \lambda w, \_, y.\mathsf{ist}_w\, y.$$

Let $\mathsf{EqvTrees}_Y$ be the subsetoid of $\widetilde{\mathsf{W}}_Y$ on the equivariant trees, that is to say, two equivariant trees are equal in $\mathsf{EqvTrees}_Y$ if they are pointwise equal.

Let also $\mathsf{EqvTrees}_Y^=$ be the subsetoid of W_Y on the equivariant trees, that is, two equivariant trees w and w' ar equal in $\mathsf{EqvTrees}_Y^=$ if $w =_{\mathsf{W}_Y} w'$.

Let $B : \mathsf{Fam}\, X$ be a discrete setoid family over the discrete setoid on $X : \mathsf{U}$. Note first that, by its very definition, for every $\xi : x =_X x'$ and every $b : B\, x$, there is $l' : x' =_X x'$ such that

$$|B|_{l'}\, (B_\xi\, b) =_{B\, x'} |B|_\xi\, b. \tag{5.3}$$

Recall that we write B_ξ for the transport of the setoid family B, and $|B|_\xi$ for the transport of the underlying type family $|B|$—the former being given by definition and the latter by elimination of the identity type. The matching relation $\mathsf{W}^{\mathsf{per}}_B$ was defined in 3.1.

Lemma 5.6. *Let $B : \mathsf{Fam}\, X$ be a discrete setoid family over the discrete setoid on $X : \mathsf{U}$.*

(1) *For every $w, w' : \mathsf{W}_{|B|}$,*

$$\mathsf{W}^{\mathsf{per}}_B\, w\, w' \longrightarrow \mathsf{ptEq}_{|B|}\, w\, w'.$$

(2) *Every extensional tree is equivariant, i.e. for every $w : \mathsf{W}_{|B|}$,*

$$\mathsf{W}^{\mathsf{per}}_B\, w\, w \longrightarrow \mathsf{isEquivariant}\, w.$$

Proof. (1) The proof is by induction on $w \equiv \mathsf{sup}\, x\, f : \mathsf{W}_{|B|}$ and the inductive hypothesis tells us that if a tree is matching with an immediate subtree $f\, b$ of w, then it is pointwise equal to $f\, b$.

Let then $w' \equiv \mathsf{sup}\, x'\, f'$ be a tree an suppose that w and w' are matching. To prove that w and w' are pointwise equal we need first to provide a proof $\xi : x =_X x'$, which is given by α in Remark 3.2. We then need to show that for every $b : B\, x$ and $b' : B\, x'$, if $B_\xi\, b \approx_{B\, x'} b'$, then the immediate subtrees $f\, b$ and $f'\, b'$ are pointwise equal. By the inductive hypothesis, it is enough to show that the trees $f\, b$ and $f'\, b'$ are matching, which holds by definition.

(2) The proof is again by induction on $w : \mathsf{W}_{|B|}$, into the type

$$\mathsf{W}^{\mathsf{per}}_B\, w\, w \longrightarrow \mathsf{isEquivariant}\, w.$$

By the inductive hypothesis and the fact that every subtree of an extensional subtree is extensional 3.5, it follows that every immediate subtree of w is equivariant. To prove that w is equivariant, it only remains to show that for every $l : x =_X x$ and every $b : B\, x$, the subtrees $f\, (|B|_l\, b)$ and $f\, b$ are pointwise equal. By (1) just proved and the fact that f is extensional 3.5, it is enough to prove that $|B|_l\, b \approx_{B\, x} b$. This is immediate by Definition 5.1 and standard properties of transport along identity proofs. □

It follows from Lemma 5.6 that there is an extensional function

$$j_B : W_B \Rightarrow \mathsf{EqvTrees}_{|B|} \tag{5.4}$$

which is the identity on the underlying trees. The next lemma proves that j_B is injective.

Lemma 5.7. *Let $B : \mathsf{Fam}\, X$ be a discrete setoid family over the discrete setoid on $X : \mathsf{U}$, and let $w, w' : \mathsf{W}_{|B|}$ be equivariant trees. Then*

$$\mathsf{ptEq}_{|B|}\, w\, w' \longrightarrow \mathsf{W}^{\mathsf{per}}_B\, w\, w'.$$

Proof. This is proven by induction on the proof E that $w \equiv \mathsf{sup}\, x\, f$ is equivariant, into the type

$$\prod_{w' : \mathsf{W}_Y} \mathsf{isEquivariant}\, w' \to \mathsf{ptEq}_{|B|}\, w\, w' \to \mathsf{W}^{\mathsf{per}}_B\, w\, w'.$$

Let then $w' \equiv \mathsf{sup}\, x'\, f'$ be equivariant and suppose that w and w' are pointwise equal. In particular, there is $\xi : x =_X x'$. So to prove that w and w' are matching it is enough to show that for every $b : B\, x$ and $b' : B\, x'$, if $B_\xi\, b \approx_{B\, x'} b'$ then the immediate subtrees $f\, b$

and $f'\,b'$ are matching. This follows by the inductive hypothesis once we know that $f'\,b'$ is equivariant, and that $f\,b$ and $f'\,b'$ are pointwise equal.

The former condition follows by Definition 5.5. For the latter we have

$$\begin{aligned} f\,b &\approx_{\mathsf{ptEq}} f'(|B|_\xi\,b) \\ &\approx_{\mathsf{ptEq}} f'\,(B_\xi\,b) \\ &\approx_{\mathsf{ptEq}} f'\,b' \end{aligned}$$

using, in order, the fact that subtrees of w and w' are pointwise equal by Definition 5.2; condition (5.3), Lemma 5.3 and equivariance of w'; and Definition 5.1, Lemma 5.3 and equivariance of w'. □

Theorem 5.8. *Let $B : \mathsf{Fam}\,X$ be a discrete setoid family over the discrete setoid on X.*

(1) *The function j_B in* (5.4) *is a bijection between the setoid of extensional trees W_B and the setoid of equivariant trees $\mathsf{EqvTrees}_{|B|}$ on the underlying type family $|B|$.*
(2) *Assuming function extensionality* (5.2)*, the setoid of extensional trees W_B on B is in bijection with the subsetoid $\mathsf{EqvTrees}^{=}_{Y}$ of the (discrete setoid on the) W-type $\mathsf{W}_{|B|}$.*

Proof. (1) This is immediate from Lemmas 5.6 and 5.7.

(2) By Lemmas 5.3 and 5.4, the discrete setoid of trees $\mathsf{W}_{|B|}$ is isomorphic to the setoid $\widetilde{\mathsf{W}}_{|B|}$ of pointwise-equal trees. Now the claim follows by (1) since the setoid of equivariant trees $\mathsf{EqvTrees}_{|B|}$ is a subsetoid of $\widetilde{\mathsf{W}}_{|B|}$ by definition. □

Even if assuming function extensionality does not seem to ensure, in general, that the setoid W_B of extensional trees with respect to a discrete setoid family B is discrete, this happens whenever the base type is a 0-type, *i.e.* a type with decidable equality.

Corollary 5.9. *Let $B : \mathsf{Fam}\,X$ be a discrete setoid family over the discrete setoid on X, and suppose that the type X is a 0-type.*

(1) *Every tree in $\mathsf{W}_{|B|}$ is extensional with respect to B. In particular, the setoid W_B of extensional trees is in bijection with the setoid $\widetilde{\mathsf{W}}_{|B|}$ of pointwise-equal trees.*
(2) *Assuming function extensionality* (5.2)*, the setoid W_B of extensional trees is in bijection with the discrete setoid on the W-type $\mathsf{W}_{|B|}$.*

Proof. We only need to prove that every tree is extensional, as all the other claims follow from Theorem 5.8.

A tree is extensional if and only if it is equivariant by Lemmas 5.6 and 5.7, and it is straightforward to prove that every tree in $\mathsf{W}_{|B|}$ is equivariant, by induction on the tree and using the assumption that X is a 0-type. □

Acknowledgements

The work described in this paper would have not been possible without the support of my supervisor Erik Palmgren and, in particular, his Coq library on setoids and setoid families. The main result of this paper was presented at the Workshop on Types, Homotopy Type Theory, and Verification, held at the Hausdorff Research Institute for Mathematics in Bonn in June 2018, and a first version was completed while I was hosted at the same Institute in July 2018. I thank the organisers of the workshop for giving me the opportunity to speak, the participants for valuable feedback and the Institute for excellent working conditions.

Financial support from the Royal Swedish Academy of Sciences and the K&A Wallenberg Foundation is also acknowledged. I am grateful to Peter Dybjer for bibliographic advice and to the anonymous referees for extremely useful comments. Prooftrees were typeset using Paul Taylor's macros package.

References

[Acz78] 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.

[Acz86] 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.

[Acz93] P. Aczel. Galois: A theory development project. A report on work in progress, for the Turin meeting on the Representation of Mathematics in Logical Frameworks, January 20–23, 1993.

[AGS17] S. Awodey, N. Gambino, and K. Sojakova. Homotopy-initial algebras in type theory. *J. ACM*, 63(6):51:1–51:45, 2017.

[BCP03] G. Barthe, V. Capretta, and O. Pons. Setoids in type theory. *J. Funct. Program.*, 13(2):261–293, 2003.

[Ber05] B. van den Berg. Inductive types and exact completion. *Annals of Pure and Applied Logic*, 134(2):95–121, 2005.

[BM18] 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.

[Bre15] L. Bressan. An extension of the Minimalist Foundation. Master's thesis, Università degli Studi di Padova, 2015.

[Dyb97] P. Dybjer. Representing inductively defined sets by wellorderings in Martin-Löf's type theory. *Theoretical Computer Science*, 176(1):329–335, 1997.

[Emm18] J. Emmenegger. W-types in setoids formalised in Coq. *GitHub repository*, 2018. Available at: `https://github.com/j-emmen/W-types-in-setoids`.

[EP20] J. Emmenegger and E. Palmgren. Exact completion and constructive theories of sets. *Journal of Symbolic Logic*, 85(2), 2020.

[GH04] 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.

[GK13] N. Gambino and J. Kock. Polynomial functors and polynomial monads. *Mathematical Proceedings of the Cambridge Philosophical Society*, 154(1):153–192, 2013.

[HS00] G. Huet and A. Saïbi. Constructive Category Theory. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays in honour of Robin Milner*. MIT press, 2000.

[Mai09] M.E. Maietti. A minimalist two-level foundation for constructive mathematics. *Annals of Pure and Applied Logic*, 160(3):319–354, 2009.

[ML82] Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy Łoś, Helmut Pfeiffer, and Klaus-Peter Podewski, editors, *Logic, Methodology and Philosophy of Science VI*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153 – 175. Elsevier, 1982.

[ML84] 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.

[MP00] I. Moerdijk and E. Palmgren. Wellfounded trees in categories. *Annals of Pure and Applied Logic*, 104(1):189–218, 2000.

[MR13] M.E. Maietti and G. Rosolini. Quotient completion for the foundation of constructive mathematics. *Logica Universalis*, 7(3):371–402, 2013.

[MR16] 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.

[NPS90] B. Nordström, K. Petersson, and J.M. Smith. *Programming in Martin-Löf's type theory. An introduction.* Oxford University Press, Oxford, 1990.

[Pal92] E. Palmgren. Type-theoretic interpretation of iterated, strictly positive inductive definitions. *Archive for Mathematical Logic*, 32(2):75–99, 1992.

[Pal12a] E. Palmgren. LCC setoids in Coq. *GitHub repository*, 2012. Available at: `https://github.com/erikhpalmgren/LCC_setoids_in_Coq`.

[Pal12b] E. Palmgren. Proof-relevance of families of setoids and identity in type theory. *Archive for Mathematical Logic*, 51(1):35–47, 2012.

[Pal19] E. Palmgren. From type theory to setoids and back. Preprint. arXiv:1909.01414, 2019.

[PS89] 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.

[PW14] E. Palmgren and O. Wilander. Constructing categories and setoids of setoids in type theory. *Logical Methods in Computer Science*, 10(3), 2014.

[RS15] E. Rijke and B. Spitters. Sets in homotopy type theory. *Mathematical Structures in Computer Science*, 25(5):1172–1202, 2015.

[UFP13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations in Mathematics*. Institute for Advanced Studies, Princeton, 2013. Available at: `http://homotopytypetheory.org/book/`.

[Vid18] J. Vidmar. *Polynomial Functors and W -Types for Groupoids*. PhD thesis, University of Leeds, 2018.