

Mobiles as a Quotient-Inductive Type via Plump Ordinals

Contents

0.1	Abstract	1
0.2	Setoid Infrastructure	2
0.2.1	Setoids	2
0.2.2	Setoid Homomorphisms	2
0.2.3	Setoid Isomorphisms	2
0.2.4	Setoid Functors	3
0.3	Raw infinitary trees and plumpness	3
0.3.1	Properties of the Plump Ordering	3
0.3.2	Mobile Elements	4
0.3.3	Mobile Equivalence	5
0.3.4	The Mobile Setoids	5
0.3.5	The Mobile Diagram	5
0.3.6	The Mobile Colimit	6
0.3.7	The quotient-polynomial functor	6
0.3.8	Cocontinuity	7
0.4	Constructing Mobiles Directly in Set	7
0.4.1	Explicit colimit constructors	7
0.4.2	The functor \tilde{F}_B on Set	8
0.4.3	Cocontinuity in Set	8
0.4.4	The initial algebra	8
0.5	Remarks on WISC	8

0.1 Abstract

This note describes a construction of infinitary mobiles as a quotient-inductive type. The construction proceeds in two stages. First, we give a setoid-based description in terms of a transfinite diagram indexed by a plump tree ordinal (T, \leq) . Second, we show that the same construction can be carried out directly in the category **Set**, assuming the availability of non-recursive quotients or colimit higher inductive types.

The key property of the index order T is that it possesses a *definable supremum*:

every family of elements indexed by I has a canonical upper bound. This avoids the need for WISC (the Weakly Initial Set of Covers), which is typically required to ensure that polynomial functors preserve colimits in exact completions. Our construction is an instance where these general-purpose axioms are unnecessary.

0.2 Setoid Infrastructure

Before presenting the construction, we establish the fundamental categorical infrastructure based on setoids.

0.2.1 Setoids

A **setoid** $S = (|S|, \approx_S)$ consists of a set $|S|$ equipped with an equivalence relation \approx_S satisfying:

- **Reflexivity:** $x \approx_S x$ for all $x \in |S|$
- **Symmetry:** $x \approx_S y$ implies $y \approx_S x$
- **Transitivity:** $x \approx_S y$ and $y \approx_S z$ implies $x \approx_S z$

Setoids provide a foundation for working with quotient types while maintaining computational content.

0.2.2 Setoid Homomorphisms

A **setoid homomorphism** $f : S \rightarrow T$ is a function $f : |S| \rightarrow |T|$ that respects the equivalence relations:

$$x \approx_S y \implies f(x) \approx_T f(y)$$

Setoid homomorphisms compose in the obvious way, and every setoid has an identity homomorphism.

Two homomorphisms $f, g : S \rightarrow T$ are **homomorphically equivalent** if for all $x \approx_S y$, we have $f(x) \approx_T g(y)$. This yields an equivalence relation on the hom-sets, making setoids into a category **Setoid**.

0.2.3 Setoid Isomorphisms

A **setoid isomorphism** $\phi : S \cong T$ consists of setoid homomorphisms $\phi : S \rightarrow T$ and $\phi^{-1} : T \rightarrow S$ such that:

$$\phi^{-1} \circ \phi \approx \text{id}_S \quad \text{and} \quad \phi \circ \phi^{-1} \approx \text{id}_T$$

where \approx denotes homomorphic equivalence. Setoid isomorphism defines an equivalence relation on setoids.

0.2.4 Setoid Functors

A **setoid functor** $F : \mathbf{Setoid} \rightarrow \mathbf{Setoid}$ assigns to each setoid S a setoid $F(S)$, and to each homomorphism $f : S \rightarrow T$ a homomorphism $F(f) : F(S) \rightarrow F(T)$, satisfying:

- **Identity:** $F(\text{id}_S) \approx \text{id}_{F(S)}$
- **Composition:** $F(g \circ f) \approx F(g) \circ F(f)$
- **Congruence:** $f \approx g$ implies $F(f) \approx F(g)$

These conditions ensure that F is a proper functor on the category of setoids.

0.3 Raw infinitary trees and plumpness

Fix a set $I \neq \emptyset$, either finite or infinitary. Define the W Type T of I -branching trees:

$$\text{leaf} : T, \quad \text{node} : (I \rightarrow T) \rightarrow T.$$

Expressed as a container, this is:

$$S_T = \mathbb{K}; \quad P_T = [\perp, I]; \quad T = W \ S_T \ P_T$$

We extend this to the plump ordinal trees over the W Type as:

$$S_Z = \mathbb{K}; \quad P_Z = [\perp, \mathbb{K}, I]; \quad Z = W \ S_Z \ P_Z$$

This is because to compare two diagrams over ordinals α and β , we must be able to join the ordinals. This is necessary for proving congruence on the reverse direction of cocontinuity. The exact W type doesn't matter except that it must have sufficient constructors to be able to cover all branching cases, have a \perp constructor (so it's not empty), and a join $_ \vee _ : Z \rightarrow Z \rightarrow Z$. This gives rise to a natural inclusion:

$$\begin{aligned} \iota_T : \quad T &\rightarrow Z \\ \iota_T : \quad \text{sup}(\perp, f) &\mapsto \text{sup}(\perp, f) \\ \iota_T : \quad \text{sup}(I, f) &\mapsto \text{sup}(I, f) \end{aligned}$$

Define the plump order, defined as an inductive-inductive type with two kinds $<, \leq : Z \rightarrow Z \rightarrow \text{Prop}$ on Z inductively:

$$\frac{\exists i. \alpha \leq f i}{\alpha < \text{sup}(s, f)} \quad \frac{\forall i. f i < \alpha}{\text{sup}(s, f) \leq \alpha} \quad \forall i \in b, \ s : s_z, \ f : p_z \ s \rightarrow z, \ \alpha : z$$

0.3.1 Properties of the Plump Ordering

The plump ordering satisfies the following key properties, all proven constructively in the Agda formalization:

Reflexivity: For all $i : Z$, we have $i \leq i$. This is proven by structural induction:

$$\frac{\forall x. f(x) < \sup(s, f)}{\sup(s, f) \leq \sup(s, f)}$$

Transitivity: The plump ordering satisfies multiple forms of transitivity, proven by mutual recursion: - \leq is transitive: $i \leq j \wedge j \leq k \implies i \leq k$ - $<$ is transitive: $i < j \wedge j < k \implies i < k$ - Mixed transitivity: $i \leq j \wedge j < k \implies i < k$ and $i < j \wedge j \leq k \implies i < k$

Well-foundedness: The strict ordering $<$ is well-founded, meaning every element is accessible under the relation. This provides the foundation for transfinite induction principles over plump trees.

Preorder Structure: The relation \leq forms a preorder on Z with reflexivity and transitivity, providing the categorical foundation for diagram indexing.

Subset Characterization: We can characterize the ordering using subset relations on the strict predecessors and successors:

For plump trees $i, j : Z$, we define: - $i \subseteq j$ (downward closure) as $\forall k : Z. k < i \implies k < j$ - $i \supseteq j$ (upward closure) as $\forall k : Z. i < k \implies j < k$

Intuitively, $i \subseteq j$ means that every strict predecessor of i is also a strict predecessor of j , while $i \supseteq j$ means that every strict successor of i is also a strict successor of j . Since we're working with well-founded trees rather than classical sets, "subset" here refers to inclusion of the strict predecessor sets in the well-founded ordering.

- **Equivalence:** $i \leq j \iff i \subseteq j$ and $i \leq j \implies j \supseteq i$

Quasi-extensionality: The plump ordering is quasi-extensional, meaning:

$$(i \leq j \wedge j \leq i) \iff (i \subseteq j \wedge j \subseteq i)$$

This property connects the ordering structure with the subset structure, providing a foundation for extensional reasoning about plump trees and enabling the definable supremum property crucial for cocontinuity.

We then define $<_T : T \rightarrow Z \rightarrow Prop$ as

$$t <_T \alpha = \iota_T t < \alpha$$

This gives us a way to bound trees by size ordinals. ## Mobile Diagram Construction

0.3.2 Mobile Elements

For each plump ordinal $\alpha : Z$, define the type $P_0(\alpha)$ of mobile elements at stage α inductively:

$$\begin{aligned}
\text{leaf} &: \forall \beta : I \rightarrow Z. P_0(\text{sup}(I, \beta)) \\
\text{node} &: \forall \beta : I \rightarrow Z. (\forall i : I. P_0(\beta(i))) \rightarrow P_0(\text{sup}(I, \beta)) \\
\text{weaken} &: \forall \alpha, \gamma : Z. \alpha \leq \gamma \rightarrow P_0(\alpha) \rightarrow P_0(\gamma)
\end{aligned}$$

The key insight is that mobile elements are not simply bounded trees, but structured terms that can be: - **Leaves** at any node stage (indexed by functions $\beta : I \rightarrow Z$) - **Nodes** with recursively defined mobile children at smaller ordinals - **Weakened** from smaller to larger stages via the plump ordering

0.3.3 Mobile Equivalence

Define the mobile equivalence relation \approx^P on mobile elements by the generators:

- **Leaf equivalence:** $\text{leaf}(\beta) \approx^P \text{leaf}(\gamma)$ for any $\beta, \gamma : I \rightarrow Z$
- **Node congruence:** $\text{node}(\beta, f) \approx^P \text{node}(\gamma, g)$ if $\forall i : I. f(i) \approx^P g(i)$
- **Permutation symmetry:** $\text{node}(\beta, f) \approx^P \text{node}(\beta \circ \pi, f \circ \pi)$ for any bijection $\pi : I \leftrightarrow I$
- **Weakening transparency:** $s \approx^P \text{weaken}(\alpha, \gamma, p, s)$ for any $p : \alpha \leq \gamma$
- **Symmetry and transitivity:** \approx^P is symmetric and transitive

This gives mobile equivalence the crucial **permutation symmetry** that distinguishes mobile categories from ordinary QIT signatures.

0.3.4 The Mobile Setoids

For each stage $\alpha : Z$, define the setoid:

$$P(\alpha) := (P_0(\alpha), \approx^P)$$

This construction ensures that each stage forms a proper setoid with reflexivity, symmetry, and transitivity of the mobile equivalence relation.

0.3.5 The Mobile Diagram

The mobile diagram structure arises from the plump ordering on ordinals. For any $p : \alpha \leq \beta$ in the plump order, we obtain a setoid homomorphism:

$$P(p) : P(\alpha) \rightarrow P(\beta)$$

defined by $P(p)(s) := \text{weaken}(\alpha, \beta, p, s)$ with congruence established by the weakening transparency of mobile equivalence.

This yields the mobile diagram:

$$P : (Z, \leq) \longrightarrow \mathbf{Setoid}$$

satisfying the functorial properties: - **Identity:** $P(\text{refl}) \approx_h \text{id}$ - **Composition:** $P(q \circ p) \approx_h P(q) \circ P(p)$

The mobile diagram provides the foundation for constructing colimits that respect both the ordinal structure and the crucial permutation symmetries needed for mobile category theory.

0.3.6 The Mobile Colimit

The colimit of the mobile diagram $P : (Z, \leq) \rightarrow \mathbf{Setoid}$ is constructed as follows:

Carrier: The colimit carrier is the dependent sum:

$$\text{Colim}_0 = \sum_{\alpha:Z} |P(\alpha)|$$

Equivalence Relation: Define the colimit equivalence relation \approx^l on Colim_0 inductively by:

- **Stage equivalence:** $(\alpha, x) \approx^l (\alpha, x')$ if $P(\alpha)[x \approx^P x']$
- **Step equivalence:** $(\alpha, x) \approx^l (\beta, P(p)(x))$ for any $p : \alpha \leq \beta$
- **Symmetry:** $s \approx^l t \implies t \approx^l s$
- **Transitivity:** $s \approx^l t \wedge t \approx^l u \implies s \approx^l u$

The Colimit Setoid:

$$\text{Colim}(P) := (\text{Colim}_0, \approx^l)$$

Cocone Structure: The canonical injections are:

$$\iota_\alpha : P(\alpha) \rightarrow \text{Colim}(P)$$

$$\iota_\alpha(x) := (\alpha, x)$$

These satisfy the cocone property: for any $p : \alpha \leq \beta$, we have $\iota_\beta \circ P(p) \approx_h \iota_\alpha$.

Universal Property: $\text{Colim}(P)$ satisfies the universal property that for any other cocone with apex A and injections $\{\kappa_\alpha : P(\alpha) \rightarrow A\}$, there exists a unique setoid homomorphism $h : \text{Colim}(P) \rightarrow A$ making all triangles commute.

0.3.7 The quotient-polynomial functor

Define an endofunctor on setoids:

$$\tilde{F}_B(X, \approx_X) := (1 + (I \rightarrow X)) / \approx_F,$$

where \approx_F is the smallest equivalence relation generated by:

$$\text{Leaf} \approx_F \text{Leaf},$$

$$\text{Node}(f) \approx_F \text{Node}(g) \text{ if } \forall i, f(i) \approx_X g(i),$$

$$\text{Node}(f) \approx_F \text{Node}(g) \text{ if } \exists \pi : I \cong I, \forall i, f(i) \approx_X g(\pi(i)).$$

0.3.8 Cocontinuity

A crucial lemma is that for any $g : I \rightarrow \text{colim} P$, if $g(i)$ is represented by (t_i, x_i) then the definable supremum

$$t^* := \text{node}(i \mapsto t_i)$$

satisfies $t_i \leq t^*$, hence

$$g = \iota_{t^*} \circ h$$

for the unique $h : I \rightarrow P(t^*)$ defined by $h(i) := P(t_i \leq t^*)(x_i)$. This supplies the factorisation needed to prove that \tilde{F}_B preserves this colimit.

0.4 Constructing Mobiles Directly in Set

In this section we reinterpret the construction internally in **Set**, using a colimit higher inductive type. No use of WISC is required because the index order (T, \leq) has a definable supremum.

0.4.1 Explicit colimit constructors

Define the colimit as a $[[\text{Higher Inductive Type}]]$ M_B with constructors:

Points:

$$\text{inc}(t, x) : M_B \quad (t : T, x : P(t)).$$

Paths:

$$\text{glue}(p, x) : \text{inc}(t, x) = \text{inc}(u, P(p)(x)) \quad (p : t \leq u).$$

No further constructors are required.

0.4.2 The functor \tilde{F}_B on Set

Define

$$\tilde{F}_B(X) := (1 + (I \rightarrow X)) / \approx,$$

where \approx is the same permutation-generated relation as in the setoid case.

0.4.3 Cocontinuity in Set

Given $g : I \rightarrow M_B$, write $g(i) = \text{inc}(t_i, x_i)$. Define $t^* := \text{node}(i \mapsto t_i)$ and

$$h(i) := P(t_i \leq t^*)(x_i).$$

Then

$$g(i) = \text{inc}(t^*, h(i))$$

up to paths generated by `glue`. This provides the factor needed for the ψ map of the cocontinuity equivalence.

0.4.4 The initial algebra

Cocontinuity yields an isomorphism

$$\tilde{F}_B(M_B) \cong \text{Colim}(\tilde{F}_B \circ P),$$

from which we obtain a structure map

$$\text{mob}_B : \tilde{F}_B(M_B) \rightarrow M_B.$$

By transfinite induction along T and the colimit universal property, one obtains a unique algebra morphism into any other \tilde{F}_B -algebra. Hence:

Theorem. M_B is the [[Initial Algebra]] for \tilde{F}_B .

0.5 Remarks on WISC

In general, proving that a polynomial or quotient-polynomial functor preserves colimits in a quotient completion requires a principle such as [[WISC]] to factor arbitrary maps through a small family of covers. In this specific case, the definable supremum provided by the plump order (T, \leq) replaces the need for any choice principle.

References