Setoids to E-categories to saturated categories

or, how Erik taught me to stop worrying and love the setoids

Peter LeFanu Lumsdaine

Stockholm University

19 November 2020
Memorial conference for Erik Palmgren

Setoids

> *A set is defined by describing exactly what must be done in order to construct an element of the set and what must be done in order to show that two elements are equal.*
>
> — Errett Bishop, *Foundations of Constructive Analysis*, 1967

Setoids

> *A set is defined by describing exactly what must be done in order to construct an element of the set and what must be done in order to show that two elements are equal.*
>
> — Errett Bishop, *Foundations of Constructive Analysis*, 1967

In classical maths: common technique for constructing sets — quotient by an equivalence relation.

Setoids

> *A set is defined by describing exactly what must be done in order to construct an element of the set and what must be done in order to show that two elements are equal.*
>
> — Errett Bishop, *Foundations of Constructive Analysis*, 1967

In classical maths: common technique for constructing sets — quotient by an equivalence relation.

In (some flavours of) constructive maths: take as definition of set.

Setoids

> *A set is defined by describing exactly what must be done in order to construct an element of the set and what must be done in order to show that two elements are equal.*
>
> — Errett Bishop, *Foundations of Constructive Analysis*, 1967

In classical maths: common technique for constructing sets — quotient by an equivalence relation.

In (some flavours of) constructive maths: take as definition of setoid.

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X \colon X \longrightarrow X \longrightarrow \mathrm{Type}$, satisfying reflexivity, symmetry, transitivity.

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X : X \longrightarrow X \longrightarrow \mathrm{Type}$, and functions witnessing reflexivity, symmetry, transitivity.

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X\colon X \longrightarrow X \longrightarrow \mathrm{Type}$, satisfying reflexivity, symmetry, transitivity.

A setoid map: a function $f : X \longrightarrow Y$, sending $\sim_X$ to $\sim_Y$.

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X : X \longrightarrow X \longrightarrow \text{Type}$, satisfying reflexivity, symmetry, transitivity.

A setoid map: a function $f : X \longrightarrow Y$, sending $\sim_X$ to $\sim_Y$.

In Bishop-style, and some type-theoretic developments: most/all mathematical structures based on setoids, not sets/types.

Advantages: clear constructive content; minimal foundational commitment.

Disadvantages: much bureacracy, boilerplate lemmas; some pitfalls; arguably alien to traditional mathematics.

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X : X \longrightarrow X \longrightarrow \text{Type}$, satisfying reflexivity, symmetry, transitivity.

A setoid map: a function $f : X \longrightarrow Y$, sending $\sim_X$ to $\sim_Y$.

In Bishop-style, and some type-theoretic developments: most/all mathematical structures based on setoids, not sets/types.

Advantages: clear constructive content; minimal foundational commitment.

Disadvantages: much bureacracy, boilerplate lemmas; some pitfalls; arguably alien to traditional mathematics.

Bureaucracy and pitfalls: "setoid hell".

Setoids in type theory

(Work in type-theoretic setting. Ignore size/universe issues.)

Definition

A setoid: a type X, together with a relation $\sim_X : X \longrightarrow X \longrightarrow \mathrm{Type}$, satisfying reflexivity, symmetry, transitivity.

A setoid map: a function $f : X \longrightarrow Y$, sending $\sim_X$ to $\sim_Y$.

In Bishop-style, and some type-theoretic developments: most/all mathematical structures based on setoids, not sets/types.

Advantages: clear constructive content; minimal foundational commitment.

Disadvantages: much bureacracy, boilerplate lemmas; some pitfalls; arguably alien to traditional mathematics.

Bureaucracy and pitfalls: "setoid hell". Erik's preferred view: Bishop's purgatorium — a staging-point to Cantor's paradise!

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;
- relations $\sim_Y \ldots$ on each Y_x?

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;
- relations $\sim_Y \ldots$ on each Y_x? between Y_x and $Y_{x'}$, for $x \sim x'$?

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;
- relations $\sim_Y \ldots$ on each Y_x? between Y_x and $Y_{x'}$, for $x \sim x'$?
- respecting $\sim_X$ somehow?

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;
- relations $\sim_Y \ldots$ on each Y_x? between Y_x and $Y_{x'}$, for $x \sim x'$?
- respecting $\sim_X$ somehow?

Several equivalent correct definitions. Also some easily-mistaken incorrect definitions. Also subtle pitfalls with using the correct definitions.

Setoid hell, concretely

Need to always carry around slightly more complicated objects; prove functions respect equality, etc.

Worst with dependently-sorted structures. What is a family of setoids indexed over a setoid?

Definition

X a setoid. A family of setoids Y indexed over X:

- for each $x : X$, a type Y_x;
- relations $\sim_Y \ldots$ on each Y_x? between Y_x and $Y_{x'}$, for $x \sim x'$?
- respecting $\sim_X$ somehow?

Several equivalent correct definitions. Also some easily-mistaken incorrect definitions. Also subtle pitfalls with using the correct definitions.

Need some guiding framework!

Setoids as a translation

Powerful organisational framework: setoids as *translation* from a more extensional type theory (with quotients) into a more intensional type theory.

$$\mathbf{ETT} \longrightarrow \mathbf{ITT}$$

(Developed by various authors; notably Maietti and Sambin's two-layer Minimalist Foundation.)

Boilerplate lemmas automatically provided by translation.

Setoids as a translation

Powerful organisational framework: setoids as *translation* from a more extensional type theory (with quotients) into a more intensional type theory.

$$\mathbf{ETT} \longrightarrow \mathbf{ITT}$$

(Developed by various authors; notably Maietti and Sambin's two-layer Minimalist Foundation.)

Boilerplate lemmas automatically provided by translation.

Compare other foundational translations:

- Double-negation translation: classical to intuitionistic logic.
- Chu construction: linear HOL to IHOL (Shulman 2018).
- Program-extraction/realisability: various logics to programming languages.

E-categories

Different kind of organisational framework: category theory.

Definition

An e-category $\mathbf{C}$:

- type of objects $\mathbf{C}_0$;
- setoids of morphisms $\mathbf{C}_1(x, y)$, for $x, y : \mathbf{C}_0$;
- identities, composition maps $\mathbf{C}_1(x, y) \times \mathbf{C}_1(y, z) \longrightarrow \mathbf{C}_1(x, z)$;
- satisfying category axioms, up to setoid equality.

E-categories

Different kind of organisational framework: category theory.

Definition

An e-category $\mathbf{C}$:

- type of objects $\mathbf{C}_0$;
- setoids of morphisms $\mathbf{C}_1(x, y)$, for $x, y : \mathbf{C}_0$;
- identities, composition maps $\mathbf{C}_1(x, y) \times \mathbf{C}_1(y, z) \longrightarrow \mathbf{C}_1(x, z)$;
- satisfying category axioms, up to setoid equality.

E-categories

Different kind of organisational framework: category theory.

Definition

An e-category $\mathbf{C}$:

- type of objects $\mathbf{C}_0$;
- setoids of morphisms $\mathbf{C}_1(x, y)$, for $x, y : \mathbf{C}_0$;
- identities, composition maps $\mathbf{C}_1(x, y) \times \mathbf{C}_1(y, z) \longrightarrow \mathbf{C}_1(x, z)$;
- satisfying category axioms, up to setoid equality.

Original motivation: organise setoid-based algebra, like classical categories organise set-based algebra.

Families of setoids revisited

Definition

X a setoid. The discrete e-category $D(X)$ on X:

- type of objects X;
- hom-setoids $(x \sim_X y)$, with trivial equality $(e \sim_{x \sim_X y} e') :\equiv 1$.

Families of setoids revisited

Definition

X a setoid. The discrete e-category $D(X)$ on X:

- type of objects X;
- hom-setoids $(x \sim_X y)$, with trivial equality $(e \sim_{x \sim_X y} e') :\equiv 1$.

Definition

A family of setoids on X is an e-functor $Y : D(X) \longrightarrow \mathbf{Setoid}$.

Families of setoids revisited

Definition

X a setoid. The discrete e-category $D(X)$ on X:

- type of objects X;
- hom-setoids $(x \sim_X y)$, with trivial equality $(e \sim_{x \sim_X y} e') \coloneqq 1$.

Definition

A family of setoids on X is an e-functor $Y : D(X) \longrightarrow \mathbf{Setoid}$.

Here and other ways: e-categories clearly useful. But: outside the image of the translation $\mathbf{ETT} \longrightarrow \mathbf{ITT}$, since objects a type not a setoid

Translation is guiding but not limiting. Again, compare other foundational translations.

HoTT pre-categories

Recall: categories in HoTT/univalent foundations (Ahrens–Kapulkin–Shulman).

Definition

A pre-category $\mathbf{C}$:

- type of objects $\mathbf{C}_0$;
- sets of morphisms $\mathbf{C}_1(x, y)$, for $x, y : \mathbf{C}_0$;
- identities, composition maps $\mathbf{C}_1(x, y) \times \mathbf{C}_1(y, z) \longrightarrow \mathbf{C}_1(x, z)$;
- satisfying category axioms, up to propositional equality.

HoTT pre-categories

Recall: categories in HoTT/univalent foundations (Ahrens–Kapulkin–Shulman).

Definition

A pre-category $\mathbf{C}$:

- type of objects $\mathbf{C}_0$;
- sets of morphisms $\mathbf{C}_1(x, y)$, for $x, y : \mathbf{C}_0$;
- identities, composition maps $\mathbf{C}_1(x, y) \times \mathbf{C}_1(y, z) \longrightarrow \mathbf{C}_1(x, z)$;
- satisfying category axioms, up to propositional equality.

(Work now in HoTT; *set* means *h-set*, etc.)

Saturation

Definition

A precat $\mathbf{C}$ is saturated (aka univalent, aka a category) if for all x, y, the canonical map $(x =_{\mathbf{C}_0} y) \longrightarrow (x \cong_{\mathbf{C}} y)$ is an equivalence.

Briefly: equality of objects is isomorphism.

Saturation

Definition

A precat $\mathbf{C}$ is saturated (aka univalent, aka a category) if for all x, y, the canonical map $(x =_{\mathbf{C}_0} y) \longrightarrow (x \cong_{\mathbf{C}} y)$ is an equivalence.

Briefly: equality of objects is isomorphism.

Classically: no precategory with non-trivial automorphisms can be saturated.

In HoTT: most natural examples saturated (by univalence); most constructions preserve saturation.

When constructions break saturation: can take Rezk-completion $\mathbf{C} \longrightarrow RC(\mathbf{C})$, adding the isos as equalities in the type of objects.

Saturation

Definition

A precat $\mathbf{C}$ is saturated (aka univalent, aka a category) if for all x, y, the canonical map $(x =_{\mathbf{C}_0} y) \longrightarrow (x \cong_{\mathbf{C}} y)$ is an equivalence.

Briefly: equality of objects is isomorphism.

Classically: no precategory with non-trivial automorphisms can be saturated.

In HoTT: most natural examples saturated (by univalence); most constructions preserve saturation.

When constructions break saturation: can take Rezk-completion $\mathbf{C} \longrightarrow RC(\mathbf{C})$, adding the isos as equalities in the type of objects.

Further variant, promoted by Voevodsky in UniMath: drop assumption that hom-types are sets.

Maximal unsaturation

E-categories are the maximally unsaturated notion.

Maximal unsaturation

E-categories are the maximally unsaturated notion.

In an e-cat, call the setoid equalities $e : f \sim_{\mathbf{C}(x,y)} g$ 2-cells.

Definition

An e-category **C** is:

- 2-saturated if equality of 2-cells is trivially true,
  i.e. each $f \sim_{\mathbf{C}(x,y)} g$ is a mere proposition;
- 1-saturated if equality of arrows is 2-cells,
  i.e. each $\sim_{\mathbf{C}(x,y)}$ is actual propositional equality;
- 0-saturated if equality of objects is isomorphism.

Maximal unsaturation

E-categories are the maximally unsaturated notion.

In an e-cat, call the setoid equalities $e : f \sim_{\mathbf{C}(x,y)} g$ 2-cells.

Definition

An e-category $\mathbf{C}$ is:

- 2-saturated if equality of 2-cells is trivially true, i.e. each $f \sim_{\mathbf{C}(x,y)} g$ is a mere proposition;
- 1-saturated if equality of arrows is 2-cells, i.e. each $\sim_{\mathbf{C}(x,y)}$ is actual propositional equality;
- 0-saturated if equality of objects is isomorphism.

Pattern: equality of each sort is "indiscernability w.r.t. higher sorts". (Cf. Tsementzis et al, saturation in FOLDS-structures.)

AKS precategories: $\geq$ 1-saturation. UniMath's precategories: 1-saturation. Saturated categories: $\geq$ 0-saturation.

Back to setoids

Definition

A setoid X is:

- 1-saturated if equality of setoid-equalities is trivially true, i.e. $\sim_X$ is proposition-valued;
- 0-saturated if equality of elements is setoid-equality, i.e. $\sim_X$ is actual propositional equality on X.

0-saturated: just a type. $\geq$ 0-saturated: a set.

Back to setoids

Definition

A setoid X is:

- 1-saturated if equality of setoid-equalities is trivially true, i.e. $\sim_X$ is proposition-valued;
- 0-saturated if equality of elements is setoid-equality, i.e. $\sim_X$ is actual propositional equality on X.

0-saturated: just a type. $\geq$ 0-saturated: a set.

Taking quotient of a setoid: like taking Rezk-completion of a category.

Back to setoids

Definition

A setoid X is:

- 1-saturated if equality of setoid-equalities is trivially true, i.e. $\sim_X$ is proposition-valued;
- 0-saturated if equality of elements is setoid-equality, i.e. $\sim_X$ is actual propositional equality on X.

0-saturated: just a type. $\geq$ 0-saturated: a set.

Taking quotient of a setoid: like taking Rezk-completion of a category.

Setoids are like categories

Working with (un-saturated) setoids: analogous to working with (un-saturated) categories — standard (unavoidably) in traditional maths!

Analogy holds up surprisingly far. E.g. bureacracy of setoid lemmas — compare ubiquitous tacit functoriality/naturality lemmas.

Setoids are like categories

Working with (un-saturated) setoids: analogous to working with (un-saturated) categories — standard (unavoidably) in traditional maths!

Analogy holds up surprisingly far. E.g. bureacracy of setoid lemmas — compare ubiquitous tacit functoriality/naturality lemmas.

Response 1

Un-saturated categories are as bad as setoids! Always work with saturated categories; take Rezk-completion whenever needed.

Response 2

Setoids are good as traditional categories! Not just a constructive hack; accept setoids as genuine part of mathematical practice.

Setoids are like categories

Working with (un-saturated) setoids: analogous to working with (un-saturated) categories — standard (unavoidably) in traditional maths!

Analogy holds up surprisingly far. E.g. bureacracy of setoid lemmas — compare ubiquitous tacit functoriality/naturality lemmas.

Response 1

Un-saturated categories are as bad as setoids! Always work with saturated categories; take Rezk-completion whenever needed.

Response 2

Setoids are good as traditional categories! Not just a constructive hack; accept setoids as genuine part of mathematical practice.

Is “setoid hell” really just “formalisation hell”?