Composable monadic GAT extensions

Jonas Frey

HoTT-UF 2024

April 2, 2024

Leuven, Belgium

Generalized Algebraic Theories

Idea: Generalized Algebraic Theories (GATs) = Algebra + Type Dependency

Example: The GAT $\mathbb{T}_{\mathsf{Gph}}$ *of graphs*

$$\begin{aligned} &\vdash O \\ x\,y : O &\vdash A(x,y) \end{aligned}$$

Example: The GAT $\mathbb{T}_{\mathsf{Cat}}$ *of categories*

$$\begin{aligned} &\vdash O \\ x\,y : O &\vdash A(x,y) \\ x : O &\vdash \mathrm{id}(x) : A(x,x) \\ x\,y\,z : O\,,\ f : A(x,y)\,,\ g : A(y,z) &\vdash g{\circ}f : A(x,z) \\ x\,y : O\,,\ f : A(x,y) &\vdash \mathrm{id}(y){\circ}f = f \\ x\,y : O\,,\ f : A(x,y) &\vdash f{\circ}\mathrm{id}(x) = f \\ w\,x\,y\,z : O\,,\ e : A(w,x)\,,\ f : A(x,y)\,,\ g : A(y,z) &\vdash (g{\circ}f){\circ}e = g{\circ}(f{\circ}e) \end{aligned}$$

Formally, GATs are given by a ordered list of *axioms*, of one of the followoing forms

- declarations of (possibly dependent) sorts
- declarations of operations
- sort equations (not present above)
- term equations

The syntactic category

The **syntactic category** $\mathcal{C}$ of a GAT $\mathbb{T}[\mathcal{C}]$ is given as follows:

- objects are **contexts**
- morphisms are **substitutions**

Example: In $\mathcal{C}[\mathbb{T}_{\mathsf{Cat}}]$ we have morphisms

- $(x\,y\,z : O\,,\ f : A(x,y)\,,\ g : A(y,z)) \xrightarrow{(x\,,\,z\,,\,g\circ f)} (x\,z : O\,,\ h : A(x,z))$
- $(x\,y\,z : O\,,\ f : A(x,y)\,,\ g : A(y,z)) \xrightarrow{(x\,,\,y\,,\,f)} (x\,y : O\,,\ f : A(x,y))$

Observation: the second morphism is a *projection*, which play a special role: the syntactic category $\mathcal{C}[\mathbb{T}]$ of a GAT is (almost) a **clan**!

Clans

Definition

A **clan** is a small category $\mathcal{T}$ with terminal object 1, equipped with a class $\mathcal{T}_\dagger \subseteq \mathsf{mor}(\mathcal{T})$ of morphisms – called **display maps** and written $\twoheadrightarrow$ – such that

1. pullbacks of display maps along all maps exist and are display maps $\begin{array}{ccc} \Delta^+ & \xrightarrow{s^+} & \Gamma^+ \\ {\scriptstyle q}\downarrow & \lrcorner & \downarrow{\scriptstyle p} \\ \Delta & \xrightarrow{s} & \Gamma \end{array}$,
2. display maps are closed under composition, and
3. isomorphisms and terminal projections $\Gamma \twoheadrightarrow 1$ are display maps.

- The syntactic category $\mathcal{C}[\mathbb{T}]$ of a GAT is a clan, where the class of display maps is generated by projections and isos.
- The clan structure on the syntactic category gives us a particularly simple way to define models:

Definition

A **model** of a clan $\mathcal{T}$ is a functor $A : \mathcal{T} \to \mathsf{Set}$ which preserves 1 and pullbacks of display-maps.

We write $\mathsf{Mod}(\mathcal{T})$ for the full subcategory of $[\mathcal{T}, \mathsf{Set}]$ on models.

Functoriality of models

Definition

A **clan morphism** $\phi : \mathcal{S} \to \mathcal{T}$ is a functor which preserves 1, display maps, and pullbacks of display maps.

- For every clan morphism $\phi : \mathcal{S} \to \mathcal{T}$, the **precomposition functor** $\phi^* : [\mathcal{T}, \mathsf{Set}] \to [\mathcal{S}, \mathsf{Set}]$ restricts to models:

$$\begin{array}{ccc} \mathsf{Mod}(\mathcal{T}) & \overset{\phi^\bullet}{\dashrightarrow} & \mathsf{Mod}(\mathcal{S}) \\ \downarrow & & \downarrow \\ [\mathcal{T}, \mathsf{Set}] & \xrightarrow{\phi^*} & [\mathcal{S}, \mathsf{Set}] \end{array} \qquad \begin{array}{ccc} \mathsf{Mod}(\mathcal{T}) & \overset{\phi_\bullet}{\dashleftarrow} & \mathsf{Mod}(\mathcal{S}) \\ \downarrow & \Nwarrow & \downarrow \\ [\mathcal{T}, \mathsf{Set}] & \xleftarrow{\phi_!} & [\mathcal{S}, \mathsf{Set}] \end{array}$$

- We have left adjoints $\phi_! \dashv \phi^*$ and $\phi_\bullet \dashv \phi^\bullet$, but $\phi_\bullet$ need **not** be the restriction of the $\phi_!$! (e.g. when the same category has two different clan structures)
- This talk is about when the adjunction $\phi_\bullet \dashv \phi^\bullet$ is **monadic**, i.e. $\mathsf{Mod}(\mathcal{T})$ is the category of algebras for the monad $T_\phi = \phi^\bullet \circ \phi_\bullet : \mathsf{Mod}(\mathcal{S}) \to \mathsf{Mod}(\mathcal{S})$.

Simply typed monadicity

Single-sorted algebraic theories

- **Single-sorted algebraic theories (SSATs)** (a.k.a. **Lawvere theories**) are theories like the theory of monoids:

$$\begin{array}{rl} & \vdash\ M \\ x\,y : M & \vdash\ x{\cdot}y : M \\ & \vdash\ e : M \\ x : M & \vdash\ x{\cdot}e = e{\cdot}x = x : M \\ x\,y\,z : M & \vdash\ (x{\cdot}y){\cdot}z = x{\cdot}(y{\cdot}z) : M \end{array}$$

- Such theories can be viewed as extensions of the theory with only one sort symbol

$$\left\{\ \vdash\ M\ \right\} \hookrightarrow \left\{ \begin{array}{rl} & \vdash\ M \\ x\,y : M & \vdash\ x{\cdot}y : M \\ & \vdash\ e : M \\ x : M & \vdash\ x{\cdot}e = e{\cdot}x = x : M \\ x\,y\,z : M & \vdash\ (x{\cdot}y){\cdot}z = x{\cdot}(y{\cdot}z) : M \end{array} \right\}$$

  and the induced adjunction is monadic.
- This is the well-known fact that SSATs correspond to finitary monads on Set.
- More generally, models of algebraic theories with a set I of sorts are monadic over Set^I.

Cancellation

More generally, **extensions of algebraic theories**, such as the extension of the theory of monoids by commutativity

$$\left\{ \begin{array}{rcl} & \vdash & M \\ x\, y : M & \vdash & x \cdot y : M \\ & \vdash & e : M \\ x : M & \vdash & x \cdot e = e \cdot x = x : M \\ x\, y\, z : M & \vdash & (x \cdot y) \cdot z = x \cdot (y \cdot z) : M \end{array} \right\} \hookrightarrow \left\{ \begin{array}{rcl} & \vdash & M \\ x\, y : M & \vdash & x \cdot y : M \\ & \vdash & e : M \\ x : M & \vdash & x \cdot e = e \cdot x = x : M \\ x\, y\, z : M & \vdash & (x \cdot y) \cdot z = x \cdot (y \cdot z) : M \\ x\, y : M & \vdash & x + y = y + x \end{array} \right\},$$

of the theory of Rings over the theory of abelian groups, are monadic, in the sense that the forgetful functors

$$\mathsf{CMon} \to \mathsf{Mon} \quad \text{and} \quad \mathsf{Ring} \to \mathsf{Ab}$$

are monadic. This is because of the following **cancellation property** for monadic functors.

> *Theorem*
>
> *Given composable functors $\mathbb{C} \xrightarrow{U} \mathbb{B} \xrightarrow{U'} \mathbb{A}$, if U' and $U' \circ U$ are monadic, and U has a left adjoint, then U is monadic.*

Monadic Extensions of GATs

Failure of composability I

Things are more complicated in the generalized algebraic case:

The forgetful functors

$$\mathsf{Cat} \to \mathsf{Gph} \qquad \text{and} \qquad \mathsf{Gph} \to \mathsf{Set}^2$$

from categories to graphs, and from graphs to pairs of sets (vertices and edges) are monadic, but their composite is not.

This can be explained by saying that the 'natural' syntactic representations by extensions of theories represent Gph differently:

$$\left\{ \begin{array}{rl} \vdash & O \\ \vdash & A \end{array} \right\} \hookrightarrow \left\{ \begin{array}{rl} \vdash & O \\ \vdash & A \\ x : A \vdash & s(x) : O \\ x : A \vdash & t(x) : O \end{array} \right\}$$

$$\left\{ \begin{array}{rl} \vdash & O \\ x\,y : O \vdash & A(x,y) \end{array} \right\} \hookrightarrow \left\{ \begin{array}{rl} \vdash & O \\ x\,y : O \vdash & A(x,y) \\ x : O \vdash & \mathrm{id}(x) : A(x,x) \\ x\,y\,z : O\,,\ f : A(x,y)\,,\ g : A(y,z) \vdash & g \circ f : A(x,z) \\ \ldots \vdash & \ldots \end{array} \right\}$$

Graphs are presented non-dependently over Set^2 (with two sorts O and A and source and target maps), but the theory of categories is an extension of the theory of graphs with a **dependent** sort of arrows.

Failure of composability II

Another way monadic extensions can fail to compose is when the first extension appears in the context of the second extension.

This can happen in GATs even without adding new sort symbols because of dependency and substitution.

$$
\left\{ \begin{array}{rl} & \vdash A \\ & \vdash B \\ y : B & \vdash C(x) \end{array} \right\}
\hookrightarrow
\left\{ \begin{array}{rl} & \vdash A \\ & \vdash B \\ y : B & \vdash C(x) \\ & \vdash b : B \end{array} \right\}
\hookrightarrow
\left\{ \begin{array}{rl} & \vdash A \\ & \vdash B \\ y : B & \vdash C(x) \\ & \vdash b : B \\ z : C(b) & \vdash f(z) : A \end{array} \right\}
$$

In the following we introduce a class of composable monadic clan extensions, where this kind of thing can't happen.

Composable monadic clan extensions

The monadicity criterion

Theorem

A clan morphism $\phi : \mathcal{S} \to \mathcal{T}$ is monadic whenever $\phi^\bullet$ **is conservative and reflects algebras** *in the sense that*

$$\begin{array}{ccc} \mathsf{Mod}(\mathcal{T}) & \xrightarrow{\phi^\bullet} & \mathsf{Mod}(\mathcal{S}) \\ \downarrow & & \downarrow \\ [\mathcal{T}, \mathsf{Set}] & \xrightarrow{\phi^*} & [\mathcal{S}, \mathsf{Set}] \end{array}$$

is a (bi)pullback in Cat.

Proof.

By Beck's theorem it's enough to show that $\phi^\bullet$ preserves $\phi^\bullet$-split coequalizers.

Consider a $\phi^\bullet$-split parallel pair $f, g : A \rightrightarrows B$ in $\mathsf{Mod}(\mathcal{T})$, let $A \rightrightarrows B \xrightarrow{c} C$ be its coequalizer in $[\mathcal{T}, \mathsf{Set}]$, and let $\phi^\bullet A \rightrightarrows \phi^\bullet B \xrightarrow{d} D$ be the split coequalizer in $\mathsf{Mod}(\mathcal{S})$. As a split coequalizer the latter is absolute, and is therefore preserved by $\mathsf{Mod}(\mathcal{S}) \hookrightarrow [\mathcal{S}, \mathsf{Set}]$. Since ϕ^* preserves colimits, we have $\phi^* C \cong D$. It follows that C is a model since the square is a bipullback. □

Examples

- The extension $\mathbb{T}_{\mathsf{Gph}} \hookrightarrow \mathbb{T}_{\mathsf{Cat}}$ fulfils the criterion: a functor $\mathcal{C}[\mathbb{T}_{\mathsf{Cat}}] \to \mathsf{Set}$ is a model whenever its restriction to $\mathcal{C}[\mathbb{T}_{\mathsf{Gph}}]$ is a model.
- $\left\{ \begin{array}{rl} & \vdash A \\ & \vdash B \\ y : B & \vdash C(x) \end{array} \right\} \hookrightarrow \left\{ \begin{array}{rl} & \vdash A \\ & \vdash B \\ y : B & \vdash C(x) \\ & \vdash b : B \end{array} \right\}$ does **not** fulfil the criterion!
- What about $\mathbb{T}_{\mathsf{2\text{-}Gph}} \hookrightarrow \mathbb{T}_{\mathsf{2\text{-}Cat}}$?

Monads with arities

Definition

Given a monad $T : \mathcal{X} \to \mathcal{X}$, a small full dense subcategory $\mathcal{C} \subseteq \mathcal{X}$ is said to be an **arity for** T if for every $X \in \mathcal{X}$, the canonical colimit $X = \mathrm{col}((\mathcal{C}/X) \to \mathcal{C} \hookrightarrow \mathcal{X})$ is preserved by $\mathcal{X} \xrightarrow{T} \mathcal{X} \hookrightarrow \widehat{\mathcal{C}}$.

The following can be viewed as a kind of inverse to our monadicity theorem.

Theorem (Nerve theorem[12])

If $\mathcal{C}$ is an arity for a monad $T : \mathcal{X} \to \mathcal{X}$, we have a (bi)pullback

$$\begin{array}{ccc} \mathcal{X}^T & \longrightarrow & \mathcal{X} \\ \big\downarrow & & \big\downarrow \\ \widehat{\mathcal{C}_T} & \longrightarrow & \widehat{\mathcal{C}} \end{array}$$

where $\mathcal{X}^T$ is the **category of T-algebras***, and $\mathcal{C}_T$ is the* **relative Kleisli category***.*

[2] M. Weber. "Familial 2-functors and parametric right adjoints". English. In: *Theory and Applications of Categories* (2007), Theorem 4.10.

[2] C. Berger, P.A. Melliès, and M. Weber. "Monads with arities and their associated theories". In: *Journal of Pure and Applied Algebra* (2012), Theorem 1.10.

Segal condition

Taking $\mathcal{X} = \mathsf{Gph}$ and $\mathcal{C} \subseteq \mathcal{X}$ to be the full subcategory of finite non-empty chains $(\bullet \to \bullet \cdots \to \bullet)$ the nerve theorem recovers the **Segal condition** characterizing categories among simplicial sets.

$$\begin{array}{ccc} \mathsf{Cat} & \longrightarrow & \mathsf{Gph} \\ \big\downarrow & & \big\downarrow \\ \widehat{\mathcal{C}_T} & \longrightarrow & \widehat{\mathcal{C}} \end{array}$$

The clan morphism $\mathcal{C}[\mathbb{T}_{\mathsf{Gph}}] \to \mathcal{C}[T_{\mathsf{Cat}}]$ gives rise to a similar square

$$\begin{array}{ccc} \mathsf{Cat} & \longrightarrow & \mathsf{Gph} \\ \big\downarrow & & \big\downarrow \\ [\mathcal{C}[T_{\mathsf{Cat}}], \mathsf{Set}] & \longrightarrow & [\mathcal{C}[T_{\mathsf{Gph}}], \mathsf{Set}] \end{array}$$

wit larger presheaf categories on the bottom: chains, we use arbitrary finite graphs.

Thank you for your attention!