

Coherences for the Container Model of Type Theory

Stefania Damato Thorsten Altenkirch

University of Nottingham, UK

Workshop on
Homotopy Type Theory/Univalent Foundations

3rd April 2024

Why do we need a container model of type theory?

For categorical semantics of (Q)IITs using containers!

Why do we need a container model of type theory?

For categorical semantics of (Q)IITs using containers!

Example 1: (Simplified) intrinsic syntax of type theory

```
data Con : Set
data Ty  : Con → Set

data Con where
  ◇ : Con
  _,_ : (Γ : Con) (A : Ty Γ) → Con
  eq : (Γ : Con) (A : Ty Γ) (B : Ty (Γ , A)) →
        ((Γ , A) , B) ≡ (Γ , Σ Γ A B)

data Ty where
  ι : (Γ : Con) → Ty Γ
  Σ : (Γ : Con) (A : Ty Γ) → Ty (Γ , A) → Ty Γ
```

Why do we need a container model of type theory?

For categorical semantics of (Q)IITs using containers!

Constructors are expressed via argument and target functors
[Altenkirch et al., 2018].

E.g. For $_,_ : (\Gamma : \mathbf{Con}) (A : \mathbf{Ty} \ \Gamma) \rightarrow \mathbf{Con}$ we have:

$$L : \mathbf{A}_1 \rightarrow \mathbf{Set}$$

$$L(C, T, e) := \sum (\Gamma : C) (T \ \Gamma)$$

$$R : \int L \rightarrow \mathbf{Set}$$

$$= (C : \mathbf{Set}) (T : C \rightarrow \mathbf{Set}) (e : C) (L(C, T, e)) \rightarrow \mathbf{Set}$$

$$R(C, T, e, \Gamma, A) := C$$

Why do we need a container model of type theory?

For categorical semantics of (Q)IITs using containers!

Constructors are expressed via argument and target functors
[[Altenkirch et al., 2018](#)].

Conjecture:

Restricting ourselves to **container functors** ensures we can only construct **strictly positive** (Q)IITs.

Prerequisite:

A general way to **express type contexts as containers**, i.e. a container model of type theory.

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma: \mathbf{Set} \triangleleft P_\Gamma: S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$\llbracket S_\Gamma \triangleleft P_\Gamma \rrbracket: \mathbf{Set} \rightarrow \mathbf{Set}$$

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma: \mathbf{Set} \triangleleft P_\Gamma: S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$[[S_\Gamma \triangleleft P_\Gamma]]: \mathbf{Set} \rightarrow \mathbf{Set}$$

- Substitutions are container morphisms

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma : \mathbf{Set} \triangleleft P_\Gamma : S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$[[S_\Gamma \triangleleft P_\Gamma]] : \mathbf{Set} \rightarrow \mathbf{Set}$$

- Substitutions are container morphisms
- The empty context is $\mathbf{1} \triangleleft \mathbf{0}$

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma : \mathbf{Set} \triangleleft P_\Gamma : S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$\llbracket S_\Gamma \triangleleft P_\Gamma \rrbracket : \mathbf{Set} \rightarrow \mathbf{Set}$$

- Substitutions are container morphisms
- The empty context is $\mathbf{1} \triangleleft \mathbf{0}$
- Types in context $\Gamma = S_\Gamma \triangleleft P_\Gamma$ are generalised containers $S_A : \mathbf{Set} \triangleleft P_A : S_A \rightarrow |\int \llbracket \Gamma \rrbracket|$, with extension functor

$$\llbracket S_A \triangleleft P_A \rrbracket : (\int \llbracket \Gamma \rrbracket) \rightarrow \mathbf{Set}.$$

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma: \mathbf{Set} \triangleleft P_\Gamma: S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$\llbracket S_\Gamma \triangleleft P_\Gamma \rrbracket: \mathbf{Set} \rightarrow \mathbf{Set}$$

- Substitutions are container morphisms
- The empty context is $\mathbf{1} \triangleleft \mathbf{0}$
- Types in context $\Gamma = S_\Gamma \triangleleft P_\Gamma$ are generalised containers $S_A: \mathbf{Set} \triangleleft P_A: S_A \rightarrow \int \llbracket \Gamma \rrbracket$, with extension functor

$$\llbracket S_A \triangleleft P_A \rrbracket: (\int \llbracket \Gamma \rrbracket) \rightarrow \mathbf{Set}.$$

- Terms of type A in context Γ are dependent natural transformations from $\llbracket \Gamma \rrbracket$ to $\llbracket A \rrbracket$:

$$\int_{X: \mathbf{Set}} (\gamma: \llbracket \Gamma \rrbracket X) \rightarrow \llbracket A \rrbracket(X, \gamma)$$

The container model (outlined in [Altenkirch and Kaposi, 2021])

- Contexts are set-containers $S_\Gamma: \mathbf{Set} \triangleleft P_\Gamma: S_\Gamma \rightarrow \mathbf{Set}$ with extension functor

$$\llbracket S_\Gamma \triangleleft P_\Gamma \rrbracket: \mathbf{Set} \rightarrow \mathbf{Set}$$

- Substitutions are container morphisms
- The empty context is $\mathbf{1} \triangleleft \mathbf{0}$
- Types in context $\Gamma = S_\Gamma \triangleleft P_\Gamma$ are generalised containers $S_A: \mathbf{Set} \triangleleft P_A: S_A \rightarrow \int \llbracket \Gamma \rrbracket$, with extension functor

$$\llbracket S_A \triangleleft P_A \rrbracket: (\int \llbracket \Gamma \rrbracket) \rightarrow \mathbf{Set}.$$

- Terms of type A in context Γ are dependent natural transformations from $\llbracket \Gamma \rrbracket$ to $\llbracket A \rrbracket$:

$$\int_{X: \mathbf{Set}} (\gamma: \llbracket \Gamma \rrbracket X) \rightarrow \llbracket A \rrbracket(X, \gamma)$$

- Context extension is given by $\Gamma.A = S_A \triangleleft P_A^X$

Presheaf model

- Contexts: $\mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$
- Substitutions: natural transformations
- Types: $(\int \Gamma)^{\text{op}} \rightarrow \mathbf{Set}$
- Terms:
$$\int_{X:\mathbf{Set}} (\gamma : \Gamma X) \rightarrow A(X, \gamma)$$
- Context extension: $\Gamma.A X$
$$= \sum (\rho : \Gamma X) (A(X, \rho))$$

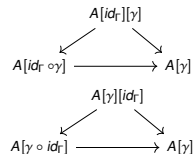
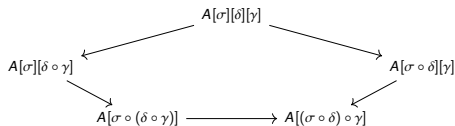
Container model

- Contexts: $\mathbf{Set} \rightarrow \mathbf{Set}$
- Substitutions: container morphisms
- Types: $(\int \llbracket \Gamma \rrbracket) \rightarrow \mathbf{Set}$
- Terms:
$$\int_{X:\mathbf{Set}} (\gamma : \llbracket \Gamma \rrbracket X) \rightarrow \llbracket A \rrbracket(X, \gamma)$$
- Context extension: $\llbracket \Gamma.A \rrbracket X$
$$= \sum (\rho : \llbracket \Gamma \rrbracket X) (\llbracket A \rrbracket(X, \rho))$$

Coherence issues in the absence of UIP

1 $Ty: \mathbf{Con}^{\text{op}} \rightarrow \mathbf{Set}$ ~~Gpd~~

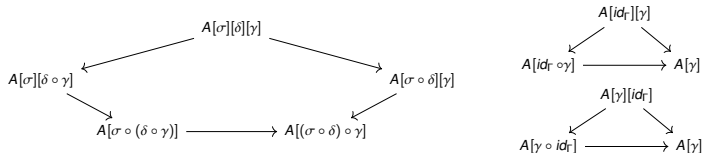
The **collection of types** is a **groupoid**, not an h-set. Additional coherence laws need to be checked.



Coherence issues in the absence of UIP

1 $Ty: \mathbf{Con}^{\text{op}} \rightarrow \mathbf{Set} \text{ Gpd}$

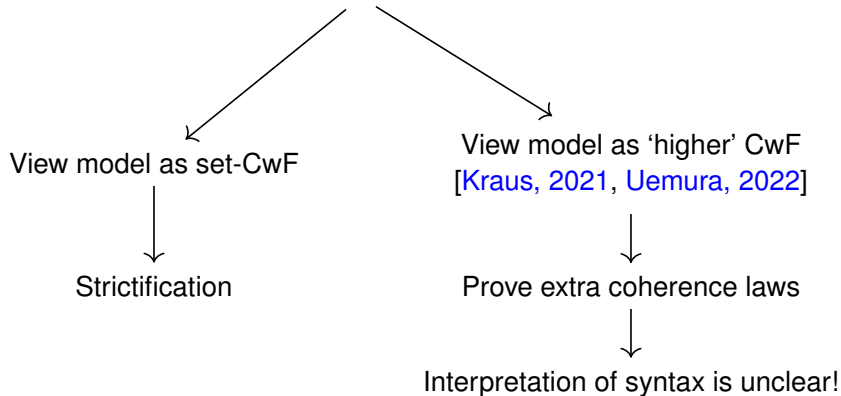
The **collection of types is a groupoid**, not an h-set. Additional coherence laws need to be checked.



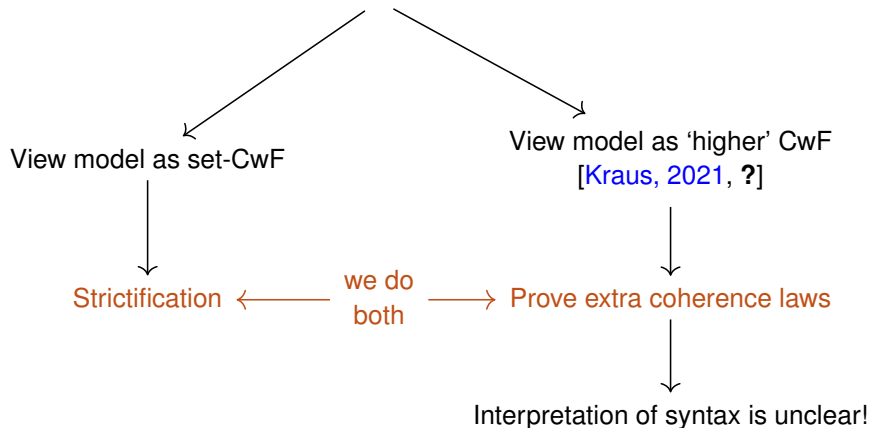
2 **Functor laws do not hold strictly**, but only up to isomorphism, due to definition of type substitution $A[\gamma]$:

$$\begin{array}{ccc}
 S_{A[\gamma]} & \xrightarrow{\text{snd}} & S_A \\
 \text{fst} \downarrow & \lrcorner & \downarrow P_A^s \\
 S_\Delta & \xrightarrow{\gamma_s} & S_\Gamma
 \end{array}
 \triangleleft
 \begin{array}{ccc}
 P_\Gamma(\gamma_s(\text{fst } s)) & \xrightarrow{P_A^f(\text{snd } s)} & P_A^X(\text{snd } s) \\
 \gamma_p(\text{fst } s) \downarrow & & \downarrow \text{inr} \\
 P_\Delta(\text{fst } s) & \xrightarrow{\text{inl}} & P_{A[\gamma]}^X s
 \end{array}$$

How do we solve the issues?



How do we solve the issues?



Strictified container model

- We use an inductive-recursive universe $U : \text{Set}, El : U \rightarrow \text{Set}$.

Strictified container model

- We use an inductive-recursive universe $U : \text{Set}, El : U \rightarrow \text{Set}$.
- Contexts are codes for set-containers

$$S_{\Gamma}^U : U \triangleleft P_{\Gamma}^U : El\ S_{\Gamma}^U \rightarrow U$$

Strictified container model

- We use an inductive-recursive universe $U : \text{Set}, El : U \rightarrow \text{Set}$.
- Contexts are codes for set-containers

$$S_\Gamma^U : U \triangleleft P_\Gamma^U : El\ S_\Gamma^U \rightarrow U$$

- Types in context Γ are codes for generalised containers, together with a substitution—we delay substitution

$$(\Gamma \xrightarrow{\delta} \Delta, S_B^U : U \triangleleft P_B^U : El\ S_B^U \rightarrow |\int[\![\Delta]\!]|^U)$$



Strictified container model

- We use an inductive-recursive universe $U : \text{Set}, El : U \rightarrow \text{Set}$.
- Contexts are codes for set-containers

$$S_\Gamma^U : U \triangleleft P_\Gamma^U : El\ S_\Gamma^U \rightarrow U$$

- Types in context Γ are codes for generalised containers, together with a substitution—we delay substitution

$$(\Gamma \xrightarrow{\delta} \Delta, S_B^U : U \triangleleft P_B^U : El\ S_B^U \rightarrow \|\int[\Delta]\|^U)$$



- Type substitution can now be defined as

$$(B[\delta])[\gamma] := B[\delta \circ \gamma]$$

Strictified container model

- We use an inductive-recursive universe $U : \text{Set}, El : U \rightarrow \text{Set}$.
- Contexts are codes for set-containers

$$S_\Gamma^U : U \triangleleft P_\Gamma^U : El\ S_\Gamma^U \rightarrow U$$

- Types in context Γ are codes for generalised containers, together with a substitution—we delay substitution

$$(\Gamma \xrightarrow{\delta} \Delta, S_B^U : U \triangleleft P_B^U : El\ S_B^U \rightarrow |\int[\![\Delta]\!]|^U)$$



- Type substitution can now be defined as

$$(B[\delta])[\gamma] := B[\delta \circ \gamma]$$

Now, the collection of types is an h-set, and functor laws hold strictly.





Contributions & ongoing work

- Worked out details of ‘higher’ container model outlined in [[Altenkirch and Kaposi, 2021](#)], including extra coherence laws
- Finalizing details of strictified container model
- Constructing Π -types, Σ -types, universe in both versions
- Started a formalisation of the ‘higher’ container model in Cubical Agda

- Worked out details of ‘higher’ container model outlined in [[Altenkirch and Kaposi, 2021](#)], including extra coherence laws
- Finalizing details of strictified container model
- Constructing Π -types, Σ -types, universe in both versions
- Started a formalisation of the ‘higher’ container model in Cubical Agda

Thank you!

References

-  Altenkirch, T., Capriotti, P., Dijkstra, G., Kraus, N., and Nordvall Forsberg, F. (2018).
Quotient inductive-inductive types.
In Baier, C. and Dal Lago, U., editors, *FoSSACS*, pages 293–310. Springer.
-  Altenkirch, T. and Kaposi, A. (2021).
A container model of type theory.
In *TYPES 2021*.
-  Kraus, N. (2021).
Internal ∞ -categorical models of dependent type theory : Towards 2l_{tt} eating hott.
Symposium on Logic in Computer Science (LICS 2021), pages 1–14.
-  Uemura, T. (2022).
Normalization and coherence for ∞ -type theories.