From Semantics to Syntax: A Type Theory for Comprehension Categories

Niyousha Najmaei, LIX, École Polytechnique
jww Benedikt Ahrens, Paige Randall North, Niels van der Weide

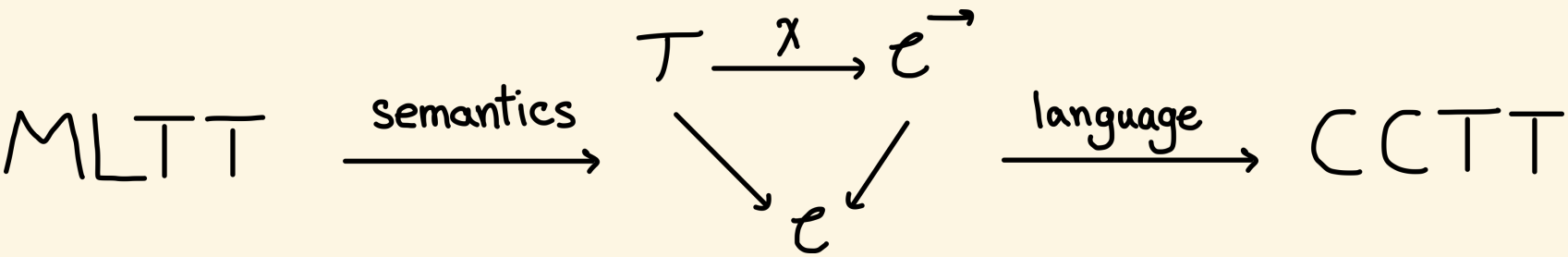
POPL, Rennes
15 Jan, 2026

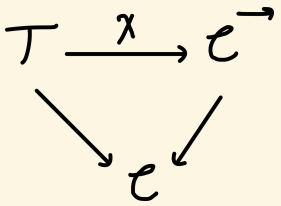
Setting

- Dependent type theory serves as a foundation for proof assistants and programming languages
- It has well-established categorical semantics: contextual categories, categories with families, display map categories, natural models, etc
- Comprehension categories are a general semantic framework for it.
  Ahrens, Lumsdaine, and North [ALN24]:
  > *"We take comprehension categories as a unifying language and show how almost all established notions of model embed as sub-2-categories (usually full) of the 2-category of comprehension categories."*

Motivation

Looking at the interpretation of Martin-Löf type theory (MLTT) in comprehension categories:



the "semantics" arrow does not use all the features of 

Two options:

1. Restrict the comprehension categories: usually done
2. Make the type theory more expressive: CCTT

Why not Restrict the Models

- Are there interesting examples we would miss?
- Are there interesting features that we would lose?

More on this after some preliminaries

Outline

1. Review: Comprehension Categories
2. Back to Our Motivation
3. Our Work: Core Syntax CCTT
4. CCTT Captures Subtyping
5. Extending CCTT with Type Formers
6. Related Work

Outline

1. Review: Comprehension Categories
2. Back to Our Motivation
3. Our Work: Core Syntax CCTT
4. CCTT Captures Subtyping
5. Extending CCTT with Type Formers
6. Related Work

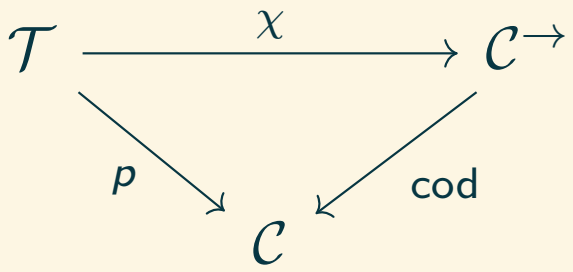
Comprehension Categories

Comprehension Category [Jac93, Definition 4.1]

A *comprehension category* consists of:

1. a category $\mathcal{C}$,
2. a (cloven) fibration $p : \mathcal{T} \to \mathcal{C}$,
3. a functor $\chi : \mathcal{T} \to \mathcal{C}^{\to}$ preserving cartesian arrows,

such that the following diagram commutes.



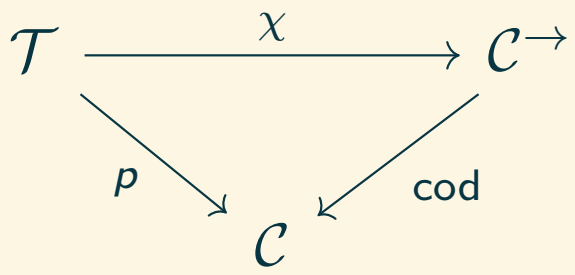
Comprehension Categories

Comprehension Category [Jac93, Definition 4.1]

A *comprehension category* consists of:

1. a category $\mathcal{C}$,
2. a (cloven) fibration $p : \mathcal{T} \to \mathcal{C}$,
3. a functor $\chi : \mathcal{T} \to \mathcal{C}^{\to}$ preserving cartesian arrows,

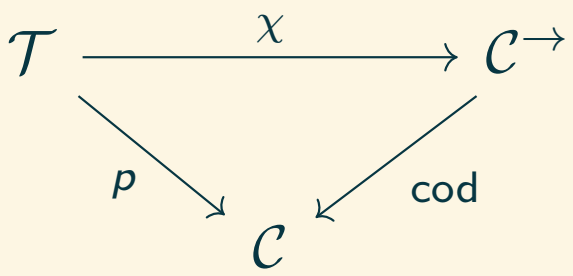
such that the following diagram commutes.



A comprehension category is *full* if χ is full and faithful; it is *split* if p is a split fibration.

Full split comprehension categories are models for MLTT.

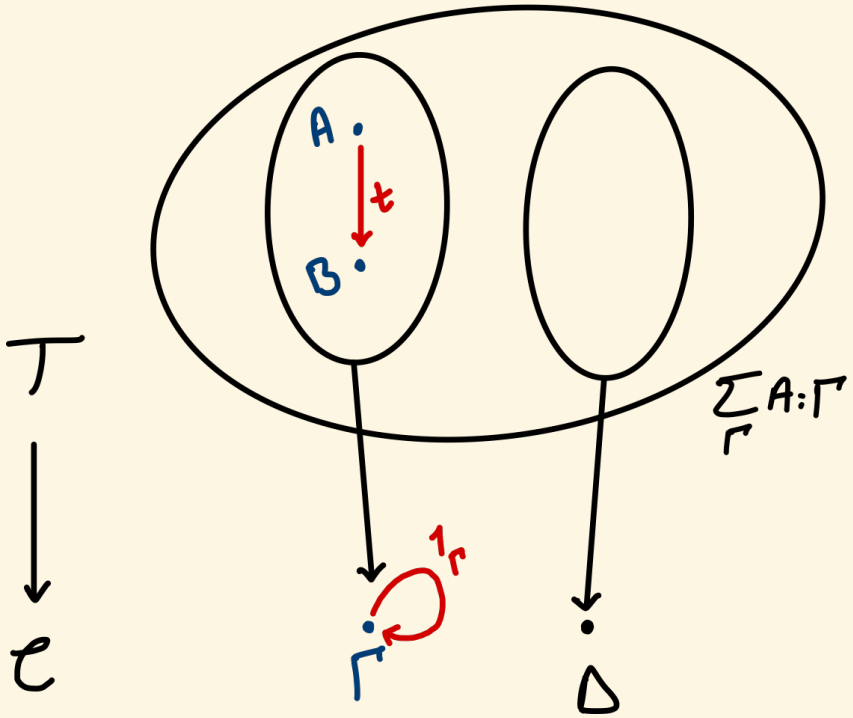
Comprehension Categories



1. $\mathcal{C}$: category of contexts and context morphisms
2. Fibre $\mathcal{T}_\Gamma$: category of types in context Γ
3. Substitution is captured by the reindexing functors
4. Extended context $\Gamma.A$ is given by $\mathrm{dom} \circ \chi : A \mapsto \Gamma.A$
5. $\Gamma \vdash t : A$ is interpreted as sections of $\chi(A) : \Gamma.A \to \Gamma$ in $\mathcal{C}$

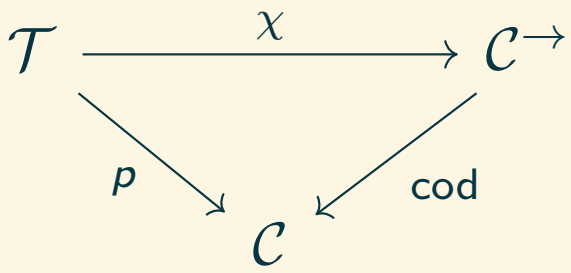
Vertical Morphisms

What about morphisms in a fibre $\mathcal{T}_\Gamma$?



Outline

Back to Our Motivation



A comprehension category can express both morphisms between contexts and morphisms between types.

Full split comprehension categories are models of MLTT: there is only one such notion there; type morphisms can be recovered from the terms of the theory.

Fullness 'kills off' this 'extra dimension' of morphisms. Later we will see that this extra dimension could capture coercive subtyping.

Interpretation of Intensional Type Theories

Are there interesting examples with this 'extra dimension' of morphisms?

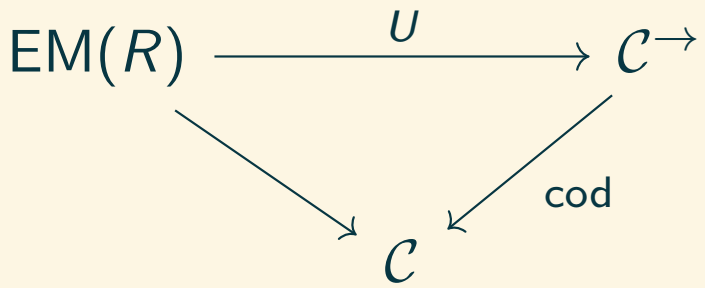
Intensional type theories, such as HoTT and Cubical are often given semantics in an algebraic weak factorisation system (AWFS) [HS98; GL23].

Interpretation of Intensional Type Theories

Are there interesting examples with this 'extra dimension' of morphisms?

Intensional type theories, such as HoTT and Cubical are often given semantics in an algebraic weak factorisation system (AWFS) [HS98; GL23].

AWFSs give rise to comprehension categories. Contrary to the ones that arise from CwFs, the comprehension categories from AWFSs are typically not full.



We capture this extra semantic structure in CCTT.

Definitional Equalities

What else does having both terms and type morphisms buy us? Tighter control over definitional equalities.

In homotopy theoretic models of MLTT from AWFSs: type morphisms are morphisms preserving transport of structure along a term of identity strictly, up to definitional equality.

$$(a = a') \to B(a) \to B(a')$$

Definitional Equalities

What else does having both terms and type morphisms buy us? Tighter control over definitional equalities.

In homotopy theoretic models of MLTT from AWFSs: type morphisms are morphisms preserving transport of structure along a term of identity strictly, up to definitional equality.

$$(a = a') \to B(a) \to B(a')$$

One can add rules to CCTT that express type morphisms strictly preserve transports, since strict preservation of these transports is validated by models of CCTT.

An example of a commonly used function in MLTT that is a type morphisms in these models: the first projection of a Σ-type.

In This Work...

1. We design rules of a type theory that reflect the structure of comprehension categories: CCTT
2. CCTT captures coercive subtyping: extends the work of Coraglia and Emmenegger [CE24]
3. Extend CCTT with Π-, Σ- and Id-types

Outline

1. Review: Comprehension Categories
2. Back to Our Motivation
3. Our Work: Core Syntax CCTT
4. CCTT Captures Subtyping
5. Extending CCTT with Type Formers
6. Related Work

CCTT: Judgements

1. Γ ctx
2. $\Gamma \vdash s : \Delta$
3. $\Gamma \vdash s \equiv s' : \Delta$
4. $\Gamma \vdash A$ type
5. $\Gamma | A \vdash t : B$
6. $\Gamma | A \vdash t \equiv t' : B$

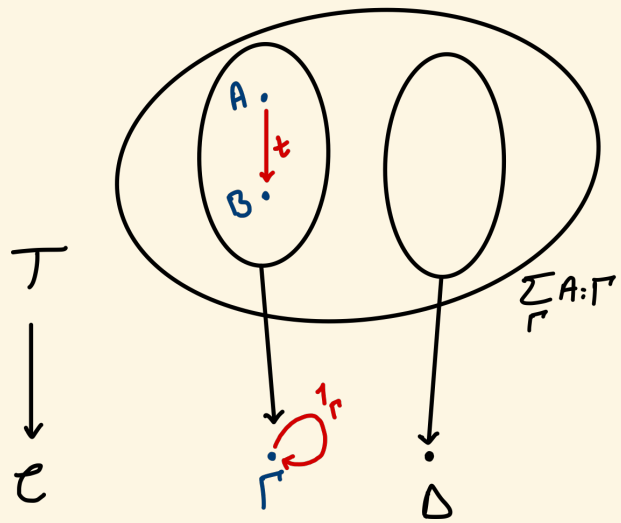
CCTT: Judgements

1. Γ ctx
2. $\Gamma \vdash s : \Delta$
3. $\Gamma \vdash s \equiv s' : \Delta$
4. $\Gamma \vdash A$ type
5. $\Gamma | A \vdash t : B$
6. $\Gamma | A \vdash t \equiv t' : B$

$\Big\}$ $\Gamma \vdash t : A$ & $\Gamma \vdash t \equiv t' : A$ in MLTT

CCTT: Judgements

1. Γ ctx
2. $\Gamma \vdash s : \Delta$
3. $\Gamma \vdash s \equiv s' : \Delta$
4. $\Gamma \vdash A$ type
5. $\Gamma | A \vdash t : B$
6. $\Gamma | A \vdash t \equiv t' : B$



Judgement 5: a morphism $[\![t]\!] : [\![A]\!] \to [\![B]\!]$ in the fibre $\mathcal{T}_{[\![\Gamma]\!]}$.

CCTT: Structural Rules

Structural rules regarding the category of contexts:

$$\frac{\Gamma \text{ ctx}}{\Gamma \vdash 1_\Gamma : \Gamma}\ \text{ctx-mor-id} \qquad \frac{\Gamma \vdash s : \Delta \quad \Delta \vdash s' : \Theta}{\Gamma \vdash s' \circ s : \Theta}\ \text{ctx-mor-comp}$$

$$\frac{\Gamma \vdash s : \Delta}{\begin{array}{c}\Gamma \vdash s \circ 1_\Gamma \equiv s : \Delta \\ \Gamma \vdash 1_\Delta \circ s \equiv s : \Delta\end{array}}\ \text{ctx-id-unit}$$

$$\frac{\Gamma \vdash s : \Delta \quad \Delta \vdash s' : \Theta \quad \Theta \vdash s'' : \Phi}{\Gamma \vdash s'' \circ (s' \circ s) \equiv (s'' \circ s') \circ s : \Phi}\ \text{ctx-comp-assoc}$$

We have similar rules for the category of types.

CCTT: Structural Rules

See the paper for the rest of the structural rules: substitution, context extension, etc

Theorem (Soundness)

Every comprehension category models the rules of CCTT.

Next, we discuss some of the rules through the lens of subtyping.

Outline

Subtyping in CCTT

Coraglia and Emmenegger [CE24] observe that the vertical morphisms can be thought of as witnesses for coercive subtyping.

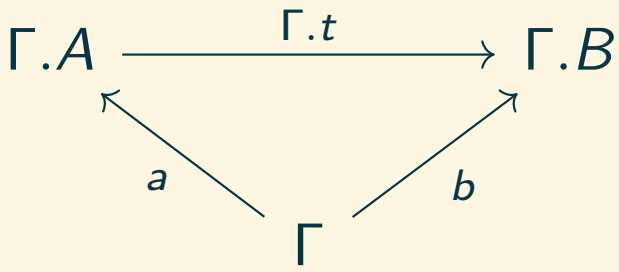
$$\Gamma | A \vdash t : B \qquad \rightsquigarrow \qquad \Gamma \vdash A \leq_t B$$

Subtyping: Subsumption

Proposition (Subsumption)

From the rules of CCTT, we can derive the following rule.

$$\frac{\Gamma \vdash A, B \text{ type} \quad \Gamma \vdash A \leq_t B \quad \Gamma \vdash a : A}{\Gamma \vdash \Gamma.t \circ a : B}$$



Γ.t is like a coercion function for $A \leq_t B$.

Subtyping: Weakening and Substitution

We have the following rule in CCTT, which corresponds to substitution for subtyping.

$$\frac{\Delta \vdash A, B \text{ type} \quad \Delta \vdash A \leq_t B \quad \Gamma \vdash s : \Delta}{\Gamma \vdash A[s] \leq_{t[s]} B[s]}$$

Proposition (Weakening for Subtyping)

From the rules of CCTT, we can derive the following rule.

$$\frac{\Gamma \vdash A, A', B \text{ type} \quad \Gamma \vdash A \leq_t A'}{\Gamma.B \vdash A[\pi_B] \leq_{t[\pi_B]} A'[\pi_{B'}]}$$

Outline

1. Review: Comprehension Categories
2. Back to Our Motivation
3. Our Work: Core Syntax CCTT
4. CCTT Captures Subtyping
5. Extending CCTT with Type Formers
6. Related Work

Subtyping for Type formers

1. Extend CCTT with a type former (e.g. Σ-types) and show soundness: naturally, no rules involving judgements of the form $\Gamma \vdash A \leq_t B$ get added.

2. Extend CCTT with subtyping for the type former and show soundness: we see how through an example!

Example: Σ-types

Extend CCTT with Σ-types, e.g.:

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\Gamma \vdash \Sigma_A B \text{ type}} \text{ sigma-form}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\Gamma.A.B \vdash \text{pair}_{\Sigma_A B} : \Gamma.\Sigma_A B} \text{ sigma-intro}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\Gamma.\Sigma_A B \vdash \text{proj}_{\Sigma_A B} : \Gamma.A.B} \text{ sigma-elim}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\begin{array}{c}\Gamma.A.B \vdash \text{proj}_{\Sigma_A B} \circ \text{pair}_{\Sigma_A B} \equiv 1_{\Gamma.A.B} : \Gamma.A.B \\ \Gamma.\Sigma_A B \vdash \text{pair}_{\Sigma_A B} \circ \text{proj}_{\Sigma_A B} \equiv 1_{\Gamma.\Sigma_A B} : \Gamma.\Sigma_A B\end{array}} \text{ sigma-beta-eta}$$

$$\frac{\Delta \vdash A \text{ type} \quad \Delta.A \vdash B \text{ type} \quad \Gamma \vdash s : \Delta}{\Gamma \mid \Sigma_{A[s]} B[s.A] \overset{\sim}{\vdash} i_{\Sigma_A B, s} : (\Sigma_A B)[s]} \text{ subst-sigma}$$

Example: Subtyping for Σ-types

1. We want to have the following rule:

$$\frac{\begin{array}{c}\Gamma \vdash A, A' \text{ type} \quad \Gamma.A \vdash B \text{ type} \quad \Gamma.A' \vdash B' \text{ type} \\ \Gamma \vdash A \leq_f A' \quad \Gamma.A \vdash B \leq_g B'[\Gamma.f]\end{array}}{\Gamma \vdash \Sigma_A B \leq_{\Sigma(f,g)} \Sigma_{A'} B'}$$

   Σ acts covariantly on both arguments.

Example: Subtyping for Σ-types

1. We want to have the following rule:

$$\frac{\Gamma \vdash A, A' \text{ type} \quad \Gamma.A \vdash B \text{ type} \quad \Gamma.A' \vdash B' \text{ type} \quad \Gamma \vdash A \leq_f A' \quad \Gamma.A \vdash B \leq_g B'[\Gamma.f]}{\Gamma \vdash \Sigma_A B \leq_{\Sigma(f,g)} \Sigma_{A'} B'}$$

   Σ acts covariantly on both arguments.
2. The coercion function for $\Sigma_A B \leq_{\Sigma(f,g)} \Sigma_{A'} B'$ should act as follows:

$$\Gamma.\Sigma_A B \xrightarrow{\text{proj}_{\Sigma_A B}} \Gamma.A.B \xrightarrow{\chi_0 g} \Gamma.A.B'[\chi_0 f] \xrightarrow{\chi_0 f.B'} \Gamma.A'.B' \xrightarrow{\text{pair}_{\Sigma_{A'} B'}} \Gamma.\Sigma_{A'} B'$$

Example: Subtyping for Σ-types

1. We want to have the following rule:

$$\frac{\begin{array}{c}\Gamma \vdash A, A' \text{ type} \quad \Gamma.A \vdash B \text{ type} \quad \Gamma.A' \vdash B' \text{ type} \\ \Gamma \vdash A \leq_f A' \quad \Gamma.A \vdash B \leq_g B'[\Gamma.f]\end{array}}{\Gamma \vdash \Sigma_A B \leq_{\Sigma(f,g)} \Sigma_{A'} B'}$$

   Σ acts covariantly on both arguments.
2. The coercion function for $\Sigma_A B \leq_{\Sigma(f,g)} \Sigma_{A'} B'$ should act as follows:

$$\Gamma.\Sigma_A B \xrightarrow{\text{proj}_{\Sigma_A B}} \Gamma.A.B \xrightarrow{\chi_0 g} \Gamma.A.B'[\chi_0 f] \xrightarrow{\chi_0 f.B'} \Gamma.A'.B' \xrightarrow{\text{pair}_{\Sigma_{A'} B'}} \Gamma.\Sigma_{A'} B'$$

3. Rules for functoriality for $\Sigma(-,-)$

Comprehension Categories with Subtyping for Σ-types

Definition

A comprehension category $(\mathcal{C}, \mathcal{T}, p, \chi)$ has subtyping for Σ-types if it has dependent sums and is equipped with a function giving for each $f : A \to A'$ in $\mathcal{T}_\Gamma$ and $g : B \to B'[\chi_0 f]$ in $\mathcal{T}_{\Gamma.A}$, a morphism

$$\Sigma_f g : \Sigma_A B \to \Sigma_{A'} B'$$

in $\mathcal{T}_\Gamma$ such that:

1. $\chi_0(\Sigma_f g)$ is the following composite

$$\Gamma.\Sigma_A B \xrightarrow{\mathrm{proj}_{\Sigma_A B}} \Gamma.A.B \xrightarrow{\chi_0 g} \Gamma.A'.B'[\chi_0 f] \xrightarrow{\chi_0 f.B'} \Gamma.A'.B' \xrightarrow{\mathrm{pair}_{\Sigma_{A'} B'}} \Gamma.\Sigma_{A'} B'$$

2. $\Sigma_{(-)}(-)$ preserves identities and composition

Theorem

Any comprehension category with subtyping for Σ-types models CCTT extended with subtyping for Σ-types.

Outline

Related Work

- Zeilberger and Melliès [MZ15] give a fibrational view of subsumptive subtyping.
- Coraglia and Emmenegger [CE24] study type morphisms as witnesses for coercive subtyping.
- Laurent, Lennon-Bertrand and Maillard [LLM24] extend MLTT to a type theory with definitionally functorial type formers and use this to extend MLTT to two type theories with coercive and subsumptive subtyping.
- Adjedj, Benjamin, Lennon-Bertrand and Maillard [Adj+25] develop a type theory modelled by split generalized categories with families and provide a general framework for defining type formers that are automatically functorial.

Summary

- We presented CCTT, which reflects the structure of a comprehension category.
- With this we gain back the 'extra dimension' of morphisms which is usually 'killed off'. This 'extra dimension' captures coercive subtyping.

Thank you for your attention!

References I

[Adj+25] Arthur Adjedj et al. "AdapTT: Functoriality for Dependent Type Casts". working paper or preprint. 2025. URL: https://hal.science/hal-05167997.

[ALN24] Benedikt Ahrens, Peter LeFanu Lumsdaine, and Paige Randall North. "Comparing Semantic Frameworks for Dependently-Sorted Algebraic Theories". In: *Programming Languages and Systems: 22nd Asian Symposium, APLAS 2024, Kyoto, Japan, October 22-24, 2024, Proceedings*. Kyoto, Japan: Springer-Verlag, 2024, pp. 3–22. ISBN: 978-981-97-8942-9. DOI: 10.1007/978-981-97-8943-6_1. URL: https://doi.org/10.1007/978-981-97-8943-6_1.

[CE24] Greta Coraglia and Jacopo Emmenegger. "Categorical Models of Subtyping". In: *29th International Conference on Types for Proofs and Programs (TYPES 2023)*. Ed. by Delia Kesner, Eduardo Hermo Reyes, and Benno van den Berg. Vol. 303. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 3:1–3:19. ISBN: 978-3-95977-332-4. DOI: 10.4230/LIPIcs.TYPES.2023.3. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.TYPES.2023.3.

[GL23] Nicola Gambino and Marco Federico Larrea. "Models of Martin-Löf Type Theory from Algebraic Weak Factorisation Systems". In: *The Journal of Symbolic Logic* 88.1 (2023), pp. 242–289. ISSN: 0022-4812,1943-5886. DOI: 10.1017/jsl.2021.39.

[HS98] Martin Hofmann and Thomas Streicher. "The Groupoid Interpretation of Type Theory". In: *Twenty-Five Years of Constructive Type Theory (Venice, 1995)*. Vol. 36. Oxford Logic Guides. New York: Oxford Univ. Press, 1998, pp. 83–111.

[Jac93] Bart Jacobs. "Comprehension Categories and the Semantics of Type Dependency". In: *Theor. Comput. Sci.* 107.2 (1993), pp. 169–207. DOI: 10.1016/0304-3975(93)90169-T. URL: https://doi.org/10.1016/0304-3975(93)90169-T.

[LLM24] Théo Laurent, Meven Lennon-Bertrand, and Kenji Maillard. "Definitional Functoriality for Dependent (Sub)Types". In: ed. by Stephanie Weirich. Vol. 14576. Lecture Notes in Computer Science. Springer, 2024, pp. 302–331. DOI: 10.1007/978-3-031-57262-3_13. URL: https://doi.org/10.1007/978-3-031-57262-3%5C_13.

References II

[MZ15] Paul-André Melliès and Noam Zeilberger. "Functors are Type Refinement Systems". In: ed. by Sriram K. Rajamani and David Walker. ACM, 2015, pp. 3–16. DOI: 10.1145/2676726.2676970. URL: https://doi.org/10.1145/2676726.2676970.