A Type Theory for Comprehension Categories with Applications to Subtyping

Niyousha Najmaei[1], Niels van der Weide[2], Benedikt Ahrens[3], Paige Randall North[4]

[1] École Polytechnique, Palaiseau, France
[2] Radboud University Nijmegen, The Netherlands
[3] Delft University of Technology, The Netherlands
[4] Utrecht University The Netherlands

We develop a type theory that we show is an internal language for comprehension categories. Usually, the semantics of Martin-Löf type theory (MLTT) is given in discrete or full comprehension categories. Requiring a comprehension category to be full or discrete can be understood as removing one 'dimension' of morphisms. In our syntax, we recover this extra dimension. We show that this extra dimension can be used naturally to encode subtyping as sketched by Coraglia and Emmenegger [5]. We then extend our type theory with Π-, Σ- and Id-types and discuss how to add subtyping for these type formers to both the syntax and semantics.

Motivation There are two primary approaches to studying denotational semantics of a type theory: one starts with syntax and later develops semantics (e.g., simply typed lambda calculus [3], MLTT [9]), while the other begins with intended semantics and then creates a syntax for it (e.g., cubical type theory [2, 4]). One can think of the latter as developing an internal language for a given categorical structure. More specifically, this can be thought of as developing a class of languages that can be used to soundly reason about the given categorical structure. Each instance of the categorical structure then gives a specific language, called its internal language.

We enact the semantics-first process by developing an internal language for comprehension categories. We focus on comprehension categories as they constitute the most general semantics for dependent type theory in the sense that all other categorical structures that are used to interpret dependent theory embed in and can be compared in comprehension categories [1]. However, the structural rules of Martin-Löf dependent type theory (strMLTT) are far from being complete with respect to comprehension categories – indeed, one often restricts to full or discrete comprehension categories to give semantics for strMLTT. Instead of restricting comprehension categories to full or discrete ones, we enlarge the syntax to describe all comprehension categories.

A comprehension category consists, among other things, of two categories: one whose objects interpret the contexts and one which interprets types. Thus, at first glance, a comprehension category can express morphisms between both contexts and types. In strMLTT, however, type morphisms can be recovered from the context morphisms. Both the requirements of discreteness and fullness 'kill off' this 'extra dimension' of morphisms. By not postulating discreteness or fullness, we gain back this 'extra dimension' in the syntax. As argued by Coraglia and Emmenegger [5], morphisms between types can be used to encode subtyping in a natural way. Thus, our syntax and semantics can capture coercive subtyping.

CCTT We design judgements and structural rules, which we call CCTT. The judgements of CCTT include the following two judgements which are not present in strMLTT.

1. $\Gamma \vdash s : \Delta$, where Γ, Δ ctx
2. $\Gamma \mid A \vdash t : B$, where $\Gamma \vdash A, B$ type

Judgement 1 adds explicit substitution to the syntax in the sense of [6]. Judgement 2 expresses 't is a witness for A being a subtype of B'. Judgements 1 and 2 are interpreted as the morphisms in the category of contexts and types in a comprehension category, respectively.

We want types (contexts) and type morphisms (substitutions) to form a category. The rules of CCTT regarding type morphisms are as follows.

$$\frac{\Gamma \ \mathsf{ctx} \quad \Gamma \vdash A \ \mathsf{type}}{\Gamma \mid A \vdash 1_A : A}\ \textsf{ty-mor-id} \qquad \frac{\Gamma \vdash A, B, C \ \mathsf{type} \quad \Gamma \mid A \vdash t : B \quad \Gamma \mid B \vdash t' : C}{\Gamma \mid A \vdash t' \circ t : C}\ \textsf{ty-mor-comp}$$

$$\frac{\Gamma \mid A \vdash t : B}{\begin{array}{l}\Gamma \mid A \vdash t \circ 1_A \equiv t : B \\ \Gamma \mid A \vdash 1_B \circ t \equiv t : B\end{array}}\ \textsf{ty-id-unit} \qquad \frac{\Gamma \mid A \vdash t : B \quad \Gamma \mid B \vdash t' : C \quad \Gamma \mid C \vdash t'' : D}{\Gamma \mid A \vdash t'' \circ (t' \circ t) \equiv (t'' \circ t') \circ t : D}\ \textsf{ty-comp-assoc}$$

We have similar rules for context morphisms. The rules for context extension and substitution mirror the action of the comprehension functor and the reindexing functors in a comprehension category, respectively. Some of these rules are as follows:

$$\frac{\Gamma \ \mathsf{ctx} \quad \Gamma \vdash A \ \mathsf{type}}{\Gamma.A \ \mathsf{ctx}}\ \textsf{ext-ty} \qquad \frac{\Gamma \vdash A, B \ \mathsf{type} \quad \Gamma \mid A \vdash t : B}{\Gamma.A \vdash \Gamma.t : \Gamma.B}\ \textsf{ext-tm} \qquad \frac{\Gamma \ \mathsf{ctx} \quad \Gamma \vdash A \ \mathsf{type}}{\Gamma.A \vdash \pi_A : \Gamma}\ \textsf{ext-proj}$$

$$\frac{\Gamma, \Delta \ \mathsf{ctx} \quad \Gamma \vdash s : \Delta \quad \Delta \vdash A \ \mathsf{type}}{\Gamma \vdash A[s] \ \mathsf{type}}\ \textsf{sub-ty} \qquad \frac{\Delta \vdash A, B \ \mathsf{type} \quad \Gamma \vdash s : \Delta \quad \Delta \mid A \vdash t : B}{\Gamma \mid A[s] \vdash t[s] : B[s]}\ \textsf{sub-tm}$$

In CCTT, the term judgement $\Gamma \vdash a : A$ is not a primitive; instead, we introduce a notation for it that stands for two judgements. This notation mirrors terms being interpreted as sections of the projection morphisms $\pi_A : \Gamma.A \to \Gamma$ in a comprehension category.

Notation 1. $\Gamma \vdash a : A$ stands for the following two judgements:

1. $\Gamma \vdash a : \Gamma.A$
2. $\Gamma \vdash \pi_A \circ a \equiv 1_\Gamma : \Gamma$

Theorem 2 (Soundness)**.** *Every comprehension category models the rules of* CCTT*.*

Subtyping One can regard type morphisms as witnesses of coercive subtyping: a judgement $\Gamma \mid A \vdash t : B$ can be seen as t is a witness for the subtyping relation $A \leq B$. Coraglia and Emmenegger explore this in generalized categories with families (GCwFs), a structure equivalent to comprehension categories [5]. In their notation, this judgement is written as $\Gamma \vdash A \leq_t B$.

Proposition 3. From the rules of CCTT, we can derive the following rule.

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

The rule in proposition 3 states that if A is a subtype of B, then a term of type A can be coerced to a term of type B. This corresponds to the subsumption rule in coercive subtyping.

In the following table, we discuss the meaning of some of the rules of CCTT from the subtyping perspective, and how they relate to the rules discussed in [5].

Rule of CCTT	Meaning under Subtyping	Rule in [5]
ty-mor-id	Reflexivity of subtyping witnessed by 1_A	-
ty-mor-comp	$A \leq_f B$ and $B \leq_g C$ give $A \leq_{g \circ f} C$.	*Trans* and *Sbsm*
ty-id-unit	Each 1_A is an identity for witness composition.	-
ty-comp-assoc	Composition of witnesses is associative.	-
ext-tm	$A \leq_t B$ gives a context morphism $\Gamma.A \vdash \Gamma.t : \Gamma.B$.	-
sub-tm	Substitution preserves subtyping.	*Wkn* and *Sbst*

Type Formers We extend CCTT with Π-, Σ- and Id-types, and show that these extensions can be interpreted in any comprehension category with suitable structure for each type former. We then discuss how CCTT can be extended with subtyping for each type former, and define a suitable semantic structure for interpreting these extensions.

Related Work Coercive and subsumptive subtyping have been studied from different angles.

Coraglia and Emmenegger [5] observe that vertical morphisms in GCwFs can be seen as witnesses for coercive subtyping. They also show this in the presence of Π- and Σ-types.

Luo and Adams [8] study structural coercive subtyping syntactically. They address coherence issues of composition of subtyping by having functoriality for type formers.

Laurent, Lennon-Bertrand and Maillard [7] extend MLTT to a type theory with functorial type formers. They use this functoriality to extend MLTT to two type theories with coercive and subsumptive subtyping, respectively. They also study meta-theoretic properties of their systems, e.g. showing that their functorial system is normalizing and has decidable type checking.

Zeilberger and Melliès [10] give a categorical view of subsumptive subtyping. They interpret type systems as functors from a category of type derivations to a category of underlying terms. In this setting, subtyping derivations are vertical morphisms.

References

[1] Benedikt Ahrens, Peter LeFanu Lumsdaine, and Paige Randall North. Comparing semantic frameworks for dependently-sorted algebraic theories. In Oleg Kiselyov, editor, *Programming Languages and Systems - 22nd Asian Symposium, APLAS 2024, Kyoto, Japan, October 22-24, 2024, Proceedings*, volume 15194 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2024.

[2] Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. In Ralph Matthes and Aleksy Schubert, editors, *19th International Conference on Types for Proofs and Programs, TYPES 2013, April 22-26, 2013, Toulouse, France*, volume 26 of *LIPIcs*, pages 107–128. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2013.

[3] Alonzo Church. A formulation of the simple theory of types. *The journal of symbolic logic*, 5(2):56–68, 1940.

[4] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: A constructive interpretation of the univalence axiom. *FLAP*, 4(10):3127–3170, 2017.

[5] Greta Coraglia and Jacopo Emmenegger. Categorical Models of Subtyping. In Delia Kesner, Eduardo Hermo Reyes, and Benno van den Berg, editors, *29th International Conference on Types for Proofs and Programs (TYPES 2023)*, volume 303 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[6] Pierre-Louis Curien, Richard Garner, and Martin Hofmann. Revisiting the categorical interpretation of dependent type theory. *Theor. Comput. Sci.*, 546:99–119, 2014.

[7] Théo Laurent, Meven Lennon-Bertrand, and Kenji Maillard. Definitional functoriality for dependent (sub)types. volume 14576 of *Lecture Notes in Computer Science*, pages 302–331. Springer, 2024.

[8] Zhaohui Luo and Robin Adams. Structural subtyping for inductive types with functorial equality rules. *Math. Struct. Comput. Sci.*, 18(5):931–972, 2008.

[9] Per Martin-Löf. *Intuitionistic type theory*, volume 1 of *Studies in proof theory*. Bibliopolis, 1984.

[10] Paul-André Melliès and Noam Zeilberger. Functors are type refinement systems. pages 3–16. ACM, 2015.