Setoid type theory*
(draft)

Ambrus Kaposi

August 14, 2018

Abstract

In this paper we turn the setoid model of type theory into a syntactic translation. More specifically, we define three type theories and translations between them. The first one is $\mathsf{MLTT}_{\mathsf{Prop}}$ which is plain intensional Martin-Löf type theory with a definitionally proof irrelevant (strict) universe of propositions. $\mathsf{MLTT}_{\mathsf{Prop}}^{\mathsf{id}+\mathsf{funext}+\mathsf{propext}}$ extends this with a propositional identity type and the axioms of functional extensionality and propositional extensionality. The third theory is SeTT, setoid type theory which has built-in syntax for working with setoids. We show how $\mathsf{MLTT}_{\mathsf{Prop}}^{\mathsf{id}+\mathsf{funext}+\mathsf{propext}}$ can be translated to $\mathsf{MLTT}_{\mathsf{Prop}}$ using a syntactic translation which follows the structure of the setoid model of type theory. This translation (unlike the original setoid model) justifies a definitional computation rule for the identity type.

A distinctive feature in SeTT is that the function space has two eliminators: one is the usual application, the other expresses preservation of equality.

We leave the treatment of a universe where identity is equality of codes as future work.

1 Introduction

Intensional type theory lacks extensionality principles such as equality of pointwise equal functions, equality of equivalent propositions or quotient types. These can be added to type theory as axioms, however this destroys its computational behaviour. A systematic study of these principles was given in Hofmann's thesis [13]. He suggests a setoid model of type theory which supports the previously mentioned concepts. In this model, a type is interpreted by a set together with an equivalence relation. The equivalence relation defines equality for the given type: for example, functions are equal if they map equal inputs to equal outputs; pairs are equal if they are pointwise equal. This is in contrast with Martin-Löf's inductive equality type with constructor refl and eliminator J where equality is given once and for all for every type. It was shown by Altenkirch [1] that the setoid model can be defined in a type-theoretic metatheory which supports a definitionally proof-irrelevant universe of propositions. Recently, support for

*This work was supported by the National Research, Development and Innovation Office – NKFIH, project number 127740.

such universes was added to Agda (to appear in version 2.6.0) allowing a full formalisation of Altenkirch's setoid model.

In this paper we turn this setoid model into a syntactic translation, and then turn this syntactic translation into a stand-alone type theory called *setoid type theory* (SeTT).

More specifically, we define the following type theories and translations.

$\mathsf{MLTT}_{\mathsf{Prop}}$ (Section 2) is a type theory with Π, Σ, Bool types and a definitionally proof-irrelevant universe of propositions closed under Π, Σ, $\top$ and $\bot$. In Section 4 we define a syntactic translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to itself which explains equality for each type former. We call this the setoid translation. For a type A, $A^{\sim}\, t_0\, t_1$ will be the proposition expressing equality of t_0 and t_1. The symbol $^{\sim}$ is an operation defined by induction on types, it is part of the translation.

The setoid translation is inspired by the setoid model of type theory [1] and the parametricity translation for dependent types [5]. This relationship is explained in Section 3.

In Section 5 we extend $\mathsf{MLTT}_{\mathsf{Prop}}$ by Martin-Löf's inductive identity type and the axioms of functional and propositional extensionality and we call this type theory $\mathsf{MLTT}_{\mathsf{Prop}}^{\mathsf{id}+\mathsf{funext}+\mathsf{propext}}$. We also extend the setoid translation to these new constructions: that is, to a translation from $\mathsf{MLTT}_{\mathsf{Prop}}^{\mathsf{id}+\mathsf{funext}+\mathsf{propext}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}$.

Finally in Section 6, we define SeTT: we start with $\mathsf{MLTT}_{\mathsf{Prop}}$ and add the specification of the previous translation as derivation rules. The implemenation of the translation is added as definitional equalities. In SeTT, the $^{\sim}$ symbol in $A^{\sim}\, t_0\, t_1$ is part of the syntax (and not an operation). Another feature of SeTT is that the function space has two eliminators. The first is application, the second expresses that the function respects equality — this is in line with the setoid model where a function is modelled by a pair: a function and a proof that it respects equality. We justify SeTT by a translation into $\mathsf{MLTT}_{\mathsf{Prop}}$ where the syntax $^{\sim}$ is modelled by the operation $^{\sim}$.

Note that we don not cover a universe of setoids, we only have a universe of propositions.

Most of Section 3 was formalised in the pre-release version of Agda (to appear as version 2.6.0) which supports a definitionally proof-irrelevant universe of propositions. The links to the formalisation are given in that section.

1.1 Related work

Observational type theory (OTT) [4] has the same motivation and goals to our work. It combines Altenkirch's setoid model [1,13] and McBride's heterogeneous equality [17] which is morally $(t \simeq t') = \Sigma(p : \mathsf{Id}_{\mathsf{U}}\, A\, B).\mathsf{Id}_B\, (\mathsf{transport}\, p\, t)\, t'$. Instead of this heterogeneous equality, we use an identity type which is defined as a logical relation following [5]. Previously we attempted to use logical relations to define a cubical type theory [3] where equality is not necessarily a proposition. This work however is unfinished: the combinatorial complexity arising from equalities between equalities so far prevents us from writing down all the computation rules for that theory. In the setoid case, this problem dissappears by truncating equality to be a proposition.

A very powerful extensionality principle is Voevodsky's univalence axiom [19]. The cubical set model of type theory [6] is a constructive model justifying

this axiom. A type theory extracted from this model is cubical type theory [9]. The relationship between the cubical set model and cubical type theory is similar to the relationship between the setoid model and setoid type theory.

Compared to cubical type theory, our work has the advantage that the computation rule for the identity type is definitional, and furthermore we have more definitional equalities: while in cubical type theory identity of Σ types is isomorphic[1] to the two pointwise identities, in our case the isomorphism is replaced by a definitional equality. The situation is similar for other type formers. These additional definitional equalities are the main motivation for Herbelin's proposal for a cubical type theory [11]. As SeTT supports uniqueness of identity proofs (Streicher's axiom K, [20]), it is incompatible with univalence.

A description of syntactic translations in general for type theory is [7]. In contrast with this work, our translations only work on intrinsic (well-typed) terms. A translation inspired by [5] for deriving computation rules from univalence is given in [21]. This work does not define a new type theory but recovers some computational power lost by adding the univalence axiom.

2 $\mathsf{MLTT}_{\mathsf{Prop}}$

The theory $\mathsf{MLTT}_{\mathsf{Prop}}$ is intensional Martin-Löf type theory with Π, Σ, Bool types and a universe of strict propositions. This means that any two elements of a proposition are definitionally equal.

In the presentation we use named variables, we consider α-equivalent terms equal and weakening is implicit. We treat this as a lightweight notation for the formal version with De Bruijn indices and explicit weakenings.

We have four sorts: contexts, types and terms and (parallel) substitutions. We stratify types into separate (predicative) levels, hence the index i for the typing judgement.

$$\vdash \Gamma \qquad \Gamma \vdash_i A \qquad \Gamma \vdash t : A \qquad \sigma : \Gamma \Rightarrow \Delta$$

We write $\Gamma \vdash t\, t' : A$ instead of $\Gamma \vdash t : A$ and $\Gamma \vdash t' : A$ and similarly for other judgements. We use an intrinsic syntax [2], that is, we only work with valid contexts, well-formed types, well-typed terms and don't consider preterms. Also, everything is quotiented by definitional equality denoted $=$. This is the same as working in a category with families [10] with extra structure, however using variable names and implicit weakenings.

We use the following naming conventions for metavariables:

- universe levels: i, j
- contexts: Γ, Δ, Θ
- types: A, B, C
- terms: t, u, v, w, a, b, c, e
- variables: x, y, z, f
- substitutions: σ, δ, ν

[1] This is a definitional isomorphism: A and B are definitionally isomorphic, if there is an $f : A \to B$, a $g : B \to A$ and $\lambda x.f\,(g\,x) = \lambda x.x$ and vice versa where $=$ is definitional equality.

Syntax for the substitution calculus:

$$\frac{}{\vdash \cdot} \qquad \frac{\vdash \Gamma \qquad \Gamma \vdash_i A}{\vdash \Gamma, x : A} \qquad \frac{(x : A) \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Delta \vdash_i A \qquad \sigma : \Gamma \Rightarrow \Delta}{\Gamma \vdash_i A[\sigma]}$$

$$\frac{\Delta \vdash t : A \qquad \sigma : \Gamma \Rightarrow \Delta}{\Gamma \vdash t[\sigma] : A[\sigma]} \qquad \frac{\vdash \Gamma}{\epsilon : \Gamma \Rightarrow \cdot} \qquad \frac{\sigma : \Gamma \Rightarrow \Delta \qquad \Gamma \vdash t : A[\sigma]}{(\sigma, x \mapsto t) : \Gamma \Rightarrow (\Delta, x : A)}$$

$$\frac{\vdash \Gamma}{\mathsf{id}_\Gamma : \Gamma \Rightarrow \Gamma} \qquad \frac{\sigma : \Theta \Rightarrow \Delta \qquad \delta : \Gamma \Rightarrow \Theta}{\sigma \circ \delta : \Gamma \Rightarrow \Delta}$$

Here we omitted most of the laws concerning substitutions. These amount to saying that contexts and substitutions form a category with a terminal object $\cdot$ and comprehension which says that substitutions $\Gamma \Rightarrow (\Delta, x : A)$ are in a natural one-to-one correspondance with substitutions $\sigma : \Gamma \Rightarrow \Delta$ and terms $\Gamma \vdash t : A[\sigma]$. We will omit substitution laws from the presentation of the rest of the type formers for brevity. However we implicitly assume that every rule comes with its obvious substitution law making everything stable under substitution. We write $t[x \mapsto u]$ for $t[(\mathsf{id}, x \mapsto u)]$ and $\mathsf{let}\, \sigma\, \mathsf{in}\, t$ for $t[\sigma]$.

Dependent function space is given by the following syntax. We use named rules for definitional equalities so that we can refer back to them easily.

$$\frac{\Gamma \vdash_i A \qquad \Gamma, x : A \vdash_i B}{\Gamma \vdash_i (x : A) \to B} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : (x : A) \to B}$$

$$\frac{\Gamma \vdash t : (x : A) \to B}{\Gamma, x : A \vdash t \mathbin{@} x : B} \qquad \Pi\beta : (\lambda x.t) \mathbin{@} x = t \qquad \Pi\eta : \lambda x.t \mathbin{@} x = t$$

We write $A \to B$ for $(x : A) \to B$ when x does not appear in B. We write $@$ for the categorical applciation rule. The usual application rule can be recovered using a substitution and we use the same $@$ notation: $t \mathbin{@} u := (t \mathbin{@} x)[x \mapsto u]$.

Σ-types are given by the following syntax.

$$\frac{\Gamma \vdash_i A \qquad \Gamma, x : A \vdash_i B}{\Gamma \vdash_i (x : A) \times B} \qquad \frac{\Gamma \vdash u : A \qquad \Gamma \vdash v : B[x \mapsto u]}{\Gamma \vdash (u, v) : (x : A) \times B}$$

$$\frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash \mathsf{pr}_0\, t : A} \qquad \frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash \mathsf{pr}_1\, t : B[x \mapsto \mathsf{pr}_0\, t]}$$

$$\Sigma\beta_0 : \mathsf{pr}_0\, (u, v) = u \qquad \Sigma\beta_1 : \mathsf{pr}_1\, (u, v) = v \qquad \Sigma\eta : (\mathsf{pr}_0\, t, \mathsf{pr}_1\, t) = t$$

We write $A \times B$ for $(x : A) \times B$ when x does not appear in B.

Booleans:

$$\frac{}{\Gamma \vdash_0 \mathsf{Bool}} \qquad \frac{}{\Gamma \vdash \mathsf{true} : \mathsf{Bool}} \qquad \frac{}{\Gamma \vdash \mathsf{false} : \mathsf{Bool}}$$

$$\frac{\Gamma, x : \mathsf{Bool} \vdash_i C \qquad \Gamma \vdash t : \mathsf{Bool} \qquad \Gamma \vdash u : C[x \mapsto \mathsf{true}] \qquad \Gamma \vdash v : C[x \mapsto \mathsf{false}]}{\Gamma \vdash \mathsf{if}\, t\, \mathsf{then}\, u\, \mathsf{else}\, v : C[x \mapsto t]}$$

$$\mathsf{Bool}\beta\mathsf{true} : \mathsf{if}\, \mathsf{true}\, \mathsf{then}\, u\, \mathsf{else}\, v = u \qquad \mathsf{Bool}\beta\mathsf{false} : \mathsf{if}\, \mathsf{false}\, \mathsf{then}\, u\, \mathsf{else}\, v = v$$

We have a universe of strict propositions. Any two elements of a proposition are equal: this is expressed by the rule irr_a.

$$\frac{}{\Gamma \vdash_{i+1} \mathsf{Prop}_i} \qquad \frac{\Gamma \vdash a : \mathsf{Prop}_i}{\Gamma \vdash_i \underline{a}} \qquad \frac{\Gamma \vdash u : \underline{a} \qquad \Gamma \vdash v : \underline{a}}{\mathsf{irr}_a : u = v}$$

This universe is closed under dependent function space, dependent sum, unit and empty types. Decoding an element of Prop is written using underline instead of El. We use a, b, c as metavariables of type Prop. Also note that the domain of the function space needs not be a proposition however needs to have the same universe level. For Π and Σ the type formation rules, constructors and destructors are overloaded.[2]

$$\frac{\Gamma \vdash_i A \qquad \Gamma, x : A \vdash b : \mathsf{Prop}_i}{\Gamma \vdash (x : A) \to b : \mathsf{Prop}_i} \qquad \frac{\Gamma, x : A \vdash t : \underline{b}}{\Gamma \vdash \lambda x.t : \underline{(x : A) \to b}}$$

$$\frac{\Gamma \vdash t : \underline{(x : A) \to b}}{\Gamma, x : A \vdash t \mathbin{@} x : \underline{b}} \qquad \pi\beta : (\lambda x.t) \mathbin{@} x = t \qquad \pi\eta : \lambda x.t \mathbin{@} x = t$$

$$\frac{\Gamma \vdash a : \mathsf{Prop}_i \qquad \Gamma, x : A \vdash b : \mathsf{Prop}_i}{\Gamma \vdash (x : a) \times b : \mathsf{Prop}_i} \qquad \frac{\Gamma \vdash u : \underline{a} \qquad \Gamma \vdash v : \underline{b}[x \mapsto u]}{\Gamma \vdash (u, v) : \underline{(x : a) \times b}}$$

$$\frac{\Gamma \vdash t : \underline{(x : a) \times b}}{\Gamma \vdash \mathsf{pr}_0\, t : \underline{a}} \qquad \frac{\Gamma \vdash t : \underline{(x : a) \times b}}{\Gamma \vdash \mathsf{pr}_1\, t : \underline{b}[x \mapsto \mathsf{pr}_0\, t]}$$

$$\sigma\beta_0 : \mathsf{pr}_0\,(u, v) = u \qquad \sigma\beta_1 : \mathsf{pr}_1\,(u, v) = v \qquad \sigma\eta : (\mathsf{pr}_0\, t, \mathsf{pr}_1\, t) = t$$

$$\frac{}{\Gamma \vdash_0 \top : \mathsf{Prop}_0} \qquad \frac{}{\Gamma \vdash \mathsf{tt} : \underline{\top}} \qquad \frac{}{\Gamma \vdash_0 \bot : \mathsf{Prop}_0} \qquad \frac{\Gamma \vdash_i C \qquad \Gamma \vdash t : \underline{\bot}}{\Gamma \vdash \mathsf{abort}\, t : C}$$

Note that the definitional proof-irrelevance has the consequence that if we have two pairs (t, u) and (t', u') which both have type $(x : A) \times \underline{b}$, then whenever $t = t'$ we get $(t, u) = (t', u')$. We will use this fact later.

3 From model to translation

Models of type theory are usually named after the interpretation of contexts. For example, in the most basic model, the set model[3], contexts are interpreted by sets. Types become families of sets, substitutions functions and terms dependent functions.

$$\frac{\vdash \Gamma}{[\![\Gamma]\!] : \mathsf{Set}} \qquad \frac{\Gamma \vdash A}{[\![A]\!] : [\![\Gamma]\!] \to \mathsf{Set}} \qquad \frac{\sigma : \Gamma \Rightarrow \Delta}{[\![\sigma]\!] : [\![\Gamma]\!] \to [\![\Delta]\!]} \qquad \frac{\Gamma \vdash t : A}{[\![t]\!] : (\gamma : [\![\Gamma]\!]) \to [\![A]\!]\,\gamma}$$

The above is the specification of the set model. Its implementation amounts to saying how each different context, type, subsitution and term formation rule is interpreted and showing that all the definitional equalities are respected. For example, context extension is interpreted by Σ types: $[\![\Gamma, x : A]\!] := (\gamma : [\![\Gamma]\!]) \times [\![A]\!]\,\gamma$. Function space is given by function space in the metatheory, abstraction and application are also interpreted by their metatheoretic counterparts. The β rule for function space comes from β for functions in the metatheory.

$$[\![(x : A) \to B]\!]\,\gamma := (\alpha : [\![A]\!]\,\gamma) \to [\![B]\!]\,(\gamma, \alpha)$$

[2] An alternative way to define Π for propositions would be to decode it to the large Π, that is, adding the rule $\underline{(x : A) \to b} = (x : A) \to \underline{b}$. However, then the translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}$ given in Section 4 wouldn't preserve this equality. For example, we would have $(\underline{(x : A) \to b})^{\sim}\, f_0\, f_1 = \top$, but also $(\underline{(x : A) \to b})^{\sim}\, f_0\, f_1 = ((x : A) \to \underline{b})^{\sim}\, f_0\, f_1 = (x_0 : |A|[0_\Gamma])(x_1 : |A|[1_\Gamma]) \to \cdots \neq \top$.

[3] Click for an Agda formalisation in the style of [2]. We have similar links for later models.

$$
\begin{array}{lll}
[\![\lambda x.t]\!]\,\gamma & := & \lambda\alpha.[\![t]\!]\,(\gamma,\alpha) \\
[\![t \mathbin{@} u]\!]\,\gamma & := & ([\![t]\!]\,\gamma)\,([\![u]\!]\,\gamma) \\
[\![\Pi\beta]\!] & : & [\![(\lambda x.t) \mathbin{@} u]\!] = \lambda\gamma.([\![\lambda x.t]\!]\,\gamma)\,([\![u]\!]\,\gamma) = \\
 & & \lambda\gamma.(\lambda\alpha.[\![t]\!]\,(\gamma,\alpha))\,([\![u]\!]\,\gamma) = \lambda\gamma.[\![t]\!]\,(\gamma,[\![u]\!]\,\gamma) = [\![t[x \mapsto u]]\!]
\end{array}
$$

The interpretation into the set model (denoted $[\![-]\!]$) can also be seen as a syntactic translation [7] from a source theory to target theory which has a Russel universe Set closed under Π and Σ (and possibly more). With this view in mind, we can write the specification of this syntactic translation as follows. We distinguish the source and target theory judgements by the $_{\mathsf{S}}$ and $_{\mathsf{T}}$ subscripts.

$$
\frac{\vdash_{\mathsf{S}} \Gamma}{\cdot \vdash_{\mathsf{T}} [\![\Gamma]\!]^0 : \mathsf{Set}} \qquad \frac{\Gamma \vdash_{\mathsf{S}} A}{\cdot \vdash_{\mathsf{T}} [\![A]\!]^0 : [\![\Gamma]\!]^0 \to \mathsf{Set}}
$$

$$
\frac{\sigma : \Gamma \Rightarrow_{\mathsf{S}} \Delta}{\cdot \vdash_{\mathsf{T}} [\![\sigma]\!]^0 : [\![\Gamma]\!]^0 \to [\![\Delta]\!]^0} \qquad \frac{\Gamma \vdash_{\mathsf{S}} t : A}{\cdot \vdash_{\mathsf{T}} [\![t]\!]^0 : (\gamma : [\![\Gamma]\!]^0) \to [\![A]\!]^0\,\gamma}
$$

All source theory judgements are interpreted by target theory term judgements: contexts become terms of type Set in the empty target context, types become terms of a function type with codomain Set etc. Source theory definitional equalities become target theory definitional equalities. This needs not be the case for models: in a model, source theory definitional equalities become metatheoretic equalities (which might not be definitional). For example, consider a situation where $_{\mathsf{S}}$ has definitional η for Σ types, and $_{\mathsf{T}}$ does not have this rule, however a propositional η rule can be proven. In this case we can define a model of $_{\mathsf{S}}$ in the metatheory $_{\mathsf{T}}$, but we are not able to define a syntactic translation from $_{\mathsf{S}}$ to $_{\mathsf{T}}$.

A different syntactic translation from $_{\mathsf{S}}$ to $_{\mathsf{T}}$ which also has a close relation to the standard model is the identity translation specified as follows.

$$
\frac{\vdash_{\mathsf{S}} \Gamma}{\vdash_{\mathsf{T}} [\![\Gamma]\!]^1} \qquad \frac{\Gamma \vdash_{\mathsf{S}} A}{[\![\Gamma]\!]^1 \vdash_{\mathsf{T}} [\![A]\!]^1} \qquad \frac{\sigma : \Gamma \Rightarrow \Delta}{[\![\sigma]\!]^1 : [\![\Gamma]\!]^1 \Rightarrow_{\mathsf{S}} [\![\Delta]\!]^1} \qquad \frac{\Gamma \vdash_{\mathsf{S}} t : A}{[\![\Gamma]\!]^1 \vdash_{\mathsf{T}} [\![t]\!]^1 : [\![A]\!]^1}
$$

Here, contexts are translated to contexts, types to types, substitutions to substitutions and terms to terms. $[\![-]\!]^0$ translates a context to an iterated Σ type (a term of type Set) which is iterated as many times as the length of the context. In contrast with this, $[\![-]\!]^1$ translates a context to a context of the same length.

The graph model of type theory models contexts by graphs (a set and a homogeneous binary relation over it), types by "dependent" graphs, substitutions by functions which respect the relations and terms by dependent functions which respect the relations. We use $[\![-]\!]_V$ and $[\![-]\!]_E$ for the vertex and edge com-

ponents of the model. The specification is the following.

$$\frac{\vdash \Gamma}{\begin{array}{l}[\![\Gamma]\!]_V : \mathsf{Set} \\ [\![A]\!]_E : [\![\Gamma]\!]_V \to [\![\Gamma]\!]_V \to \mathsf{Set}\end{array}}$$

$$\frac{\Gamma \vdash A}{\begin{array}{l}[\![A]\!]_V : [\![\Gamma]\!]_V \to \mathsf{Set} \\ [\![A]\!]_E : \forall \gamma_0\, \gamma_1 . [\![\Gamma]\!]_E\, \gamma_0\, \gamma_1 \to [\![A]\!]_V\, \gamma_0 \to [\![A]\!]_V\, \gamma_1 \to \mathsf{Set}\end{array}}$$

$$\frac{\sigma : \Gamma \Rightarrow \Delta}{\begin{array}{l}[\![\sigma]\!]_V : [\![\Gamma]\!]_V \to [\![\Delta]\!]_V \\ [\![\sigma]\!]_E : \forall \gamma_0\, \gamma_1 . [\![\Gamma]\!]_E\, \gamma_0\, \gamma_1 \to [\![\Delta]\!]_E\, ([\![\sigma]\!]_V\, \gamma_0)\, ([\![\sigma]\!]_V\, \gamma_1)\end{array}}$$

$$\frac{\Gamma \vdash t : A}{\begin{array}{l}[\![t]\!]_V : (\gamma : [\![\Gamma]\!]_V) \to [\![A]\!]_V\, \gamma \\ [\![t]\!]_E : \forall \gamma_0\, \gamma_1 . (\gamma_{01} : [\![\Gamma]\!]_E\, \gamma_0\, \gamma_1) \to [\![A]\!]_E\, \gamma_{01}\, ([\![t]\!]_V\, \gamma_0)\, ([\![t]\!]_V\, \gamma_1)\end{array}}$$

Just as the set model can be viewed as a syntactic translation $[\![-]\!]^0$, the graph model also has a syntactic translation variant ^{0} which simply makes explicit that the above judgements are in a target type theory in the empty context.

It is not clear how to define the ^{1} variant of the graph syntactic translation. A context should be interpreted by a context but also a binary relation over the context. There is no notion of binary relation over a context in type theory. However the equivalence of indexed families and display maps [8, p. 221] comes to rescue: a graph can also be given by a set of vertices, a set of edges and two functions *dom* and *cod* which give the endpoints of the edges. This variant of the graph model is specified as follows.

$$\frac{\vdash \Gamma}{\begin{array}{ll}[\![\Gamma]\!]_V & : \mathsf{Set} \\ [\![\Gamma]\!]_E & : \mathsf{Set} \\ [\![\Gamma]\!]_{dom} & : [\![\Gamma]\!]_E \to [\![\Gamma]\!]_V \\ [\![\Gamma]\!]_{cod} & : [\![\Gamma]\!]_E \to [\![\Gamma]\!]_V\end{array}} \qquad \frac{\sigma : \Gamma \Rightarrow \Delta}{\begin{array}{ll}[\![\sigma]\!]_V & : [\![\Gamma]\!]_V \to [\![\Delta]\!]_V \\ [\![\sigma]\!]_E & : [\![\Gamma]\!]_E \to [\![\Delta]\!]_E \\ [\![\sigma]\!]_{dom} & : [\![\sigma]\!]_V \circ [\![\Gamma]\!]_{dom} = [\![\Delta]\!]_{dom} \circ [\![\sigma]\!]_E \\ [\![\sigma]\!]_{cod} & : [\![\sigma]\!]_V \circ [\![\Gamma]\!]_{cod} = [\![\Delta]\!]_{cod} \circ [\![\sigma]\!]_E\end{array}}$$

$$\frac{\Gamma \vdash A}{\begin{array}{l}[\![A]\!]_V : [\![\Gamma]\!]_V \to \mathsf{Set} \\ [\![A]\!]_E : (\gamma_E : [\![\Gamma]\!]_E) \to [\![A]\!]_V\, ([\![\Gamma]\!]_{dom}\, \gamma_E) \to [\![A]\!]_V\, ([\![\Gamma]\!]_{cod}\, \gamma_E) \to \mathsf{Set}\end{array}}$$

$$\frac{\Gamma \vdash t : A}{\begin{array}{l}[\![t]\!]_V : (\gamma : [\![\Gamma]\!]_V) \to [\![A]\!]_V\, \gamma \\ [\![t]\!]_E : (\gamma_E : [\![\Gamma]\!]_E) \to [\![A]\!]_E\, \gamma_E\, \big([\![t]\!]_V\, ([\![\Gamma]\!]_{dom}\, \gamma_E)\big)\, \big([\![t]\!]_V\, ([\![\Gamma]\!]_{cod}\, \gamma_E)\big)\end{array}}$$

The syntactic translation corresponding to this model is the parametricity trans-

lation (logical relation translation) of Bernardy et al [5]. It is specified as follows.

$$\frac{\vdash_{\mathsf{S}} \Gamma}{\begin{array}{l}\vdash_{\mathsf{T}} [\![\Gamma]\!]_V \\ \vdash_{\mathsf{T}} [\![\Gamma]\!]_E \\ [\![\Gamma]\!]_{dom} \;: [\![\Gamma]\!]_E \Rightarrow_{\mathsf{T}} [\![\Gamma]\!]_V \\ [\![\Gamma]\!]_{cod} \;: [\![\Gamma]\!]_E \Rightarrow_{\mathsf{T}} [\![\Gamma]\!]_V\end{array}} \qquad \frac{\sigma : \Gamma \Rightarrow_{\mathsf{S}} \Delta}{\begin{array}{l}[\![\sigma]\!]_V \;: [\![\Gamma]\!]_V \Rightarrow_{\mathsf{T}} [\![\Delta]\!]_V \\ [\![\sigma]\!]_E \;: [\![\Gamma]\!]_E \Rightarrow_{\mathsf{T}} [\![\Delta]\!]_E \\ [\![\sigma]\!]_{dom} : [\![\sigma]\!]_V \circ [\![\Gamma]\!]_{dom} = [\![\Delta]\!]_{dom} \circ [\![\sigma]\!]_E \\ [\![\sigma]\!]_{cod} \;: [\![\sigma]\!]_V \circ [\![\Gamma]\!]_{cod} = [\![\Delta]\!]_{cod} \circ [\![\sigma]\!]_E\end{array}}$$

$$\frac{\Gamma \vdash_{\mathsf{S}} A}{\begin{array}{l}[\![\Gamma]\!]_V \vdash_{\mathsf{T}} [\![A]\!]_V \\ [\![\Gamma]\!]_E, x_0 : [\![A]\!]_V[[\![\Gamma]\!]_{dom}], x_1 : [\![A]\!]_V[[\![\Gamma]\!]_{cod}] \vdash_{\mathsf{T}} [\![A]\!]_E\, x_0\, x_1\end{array}}$$

$$\frac{\Gamma \vdash_{\mathsf{S}} t : A}{\begin{array}{l}[\![\Gamma]\!]_V \vdash_{\mathsf{T}} [\![t]\!]_V : [\![A]\!]_V \\ [\![\Gamma]\!]_E \vdash_{\mathsf{T}} [\![t]\!]_E : ([\![A]\!]_E\, x_0\, x_1)\big[x_0 \mapsto [\![t]\!]_V[[\![\Gamma]\!]_{dom}],\, x_1 \mapsto [\![t]\!]_V[[\![\Gamma]\!]_{cod}]\big]\end{array}}$$

Contexts become two contexts (vertices and edges) and two subsitutions (domain and codomain), substitutions become two substitutions and two equalities corresponding to the equalities in the model. $\circ$ is now composition of substitutions, while in the model it was function composition. In the model, $[\![A]\!]_E$ was a family indexed over three components, now it is a type where the three components are in the context. The function application $[\![A]\!]_V\,([\![\Gamma]\!]_{dom}\,\gamma_E)$ becomes instantiation of the substitution $[\![A]\!]_V[[\![\Gamma]\!]_{dom}]$.

In fact, this translation works even when S and T are the same, and in this case there is no need for the $_V$ components, these are just identity, i.e. $[\![\Gamma]\!]_V = \Gamma$, $[\![A]\!]_V = A$, $[\![\sigma]\!]_V = \sigma$ and $[\![t]\!]_V = t$. The unary variant of this translation was formalised in Agda.

In the Section 4 we will define a similar translation for the setoid model [1].

4 A translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}$

In the setoid model [1], a context is given by a set together with a proof-irrelevent equivalence relation. We think about this relation as propositonal equality.

$$\frac{\vdash \Gamma}{\begin{array}{l}|\Gamma| : \mathsf{Set} \\ \Gamma^{\sim} : |\Gamma| \to |\Gamma| \to \mathsf{Prop} \\ \mathsf{R}_\Gamma : (\gamma : |\Gamma|) \to \Gamma^{\sim}\,\gamma\,\gamma \\ \mathsf{S}_\Gamma : \Gamma^{\sim}\,\gamma_0\,\gamma_1 \to \Gamma^{\sim}\,\gamma_1\,\gamma_0 \\ \mathsf{T}_\Gamma : \Gamma^{\sim}\,\gamma_0\,\gamma_1 \to \Gamma^{\sim}\,\gamma_1\,\gamma_2 \to \Gamma^{\sim}\,\gamma_0\,\gamma_2\end{array}}$$

We reformulate this as a syntactic translation using the indexed family – display map equivalence as described in Section 3. The full specification is given in Subsection 4.1. Given a context Γ, we translate it to a context $|\Gamma|$, and instead of a relation we have a context Γ^{01} which is the total space of the relation. There are two projections from Γ^{01} to $|\Gamma|$, these are given by substitutions 0_Γ and 1_Γ. Reflexivity is a substitution $\mathsf{R}_\Gamma : |\Gamma| \Rightarrow \Gamma^{01}$ such that when postcomposed with the projections it results in identity. Symmetry is a substitution $\Gamma^{01} \Rightarrow \Gamma^{01}$ such that when postcomposed with one projection, it results in the other projection. We use implicit quantification over γ_0 and γ_1 for readability. We will follow this

notation later as well. Transitivity needs two substitutions into Γ^{01} which match at their second and first projections, respectively. It produces a substitution into Γ^{01} which gives the correct result when postcomposed with projections. The fact that $\Gamma^\sim$ results in Prop is reflected by the irr_Γ property: whenever there are two substitutions into Γ^{01} which match at their first and second projections, they are definitionally equal.

A type in the setoid model is given by the following components.

$$\frac{}{\Gamma \vdash A}$$

$$\begin{array}{ll}
|A| & : |\Gamma| \to \mathsf{Set} \\
A^\sim & : \Gamma^\sim\,\gamma_0\,\gamma_1 \to |A|\,\gamma_0 \to |A|\,\gamma_1 \to \mathsf{Prop} \\
\mathsf{R}_A & : (\alpha : |A|\,\gamma) \to A^\sim\,(\mathsf{R}_\Gamma\,\gamma)\,\alpha\,\alpha \\
\mathsf{S}_A & : A^\sim\,\gamma_{01}\,\alpha_0\,\alpha_1 \to A^\sim\,(\Gamma^\sim\,\gamma_{01})\,\alpha_1\,\alpha_0 \\
\mathsf{T}_A & : A^\sim\,\gamma_{01}\,\alpha_0\,\alpha_1 \to A^\sim\,\gamma_{12}\,\alpha_1\,\alpha_2 \to A^\sim\,(\mathsf{T}_\Gamma\,\gamma_{01}\,\gamma_{12})\,\alpha_0\,\alpha_2 \\
\mathsf{coe}_A & : \Gamma^\sim\,\gamma_0\,\gamma_1 \to |A|\,\gamma_0 \to |A|\,\gamma_1 \\
\mathsf{coh}_A & : (\gamma_{01} : \Gamma^\sim\,\gamma_0\,\gamma_1) \to (\alpha_0 : |A|\,\gamma_0) \to A^\sim\,\gamma_{01}\,\alpha_0\,(\mathsf{coe}_A\,\gamma_{01}\,\alpha_0)
\end{array}$$

There is a family of sets over $|\Gamma|$ and a heterogeneous equality relation which has dependent variants of the equivalence relation properties. In addition, there is a function coe which lets us coerce between the same type at equal interpretations of the context. The last component coh makes sure that coercion respects the relation for A. Reformulating these components as syntactic translations is fairly straightforward: the dependencies are given as extra components in the context. $A^\sim\,x_0\,x_1$ is a proposition in a context Γ^{01} extended with the two copies of $|A|$ substituted by the two projections from Γ^{01} to $|\Gamma|$. Reflexivity, symmetry, coercion and coherence correspond similarly directly to their counterparts in the model. Transitivity needs more care, here we need two substitutions into Γ^{01} which match when postcomposed by projections, three terms of the corresponding types and two heterogeneous equalities between these terms. We also include an extra property coeR which says that coercion substituted by reflexivity is the identity. This can also be included in the setoid model leading to an elimination rule for the identity type with a definitional computation rule. In an intensional metatheory, this setoid model needs functional extensionality in the metatheory for the function space. However, when reformulating the model as a syntactic translation, this requirement goes away, since functional extensionality is true for definitional equality: it comes from the η for function space.

In the type of coh, $A^\sim\,x_0\,(\mathsf{coe}_A\,x_0)$ is an abbreviation for $(A^\sim\,x_0\,x_1)[x_1 \mapsto \mathsf{coe}_A\,x_0]$. This is similar to how we abbreviated $(t \mathbin{@} x)[x \mapsto u]$ by $t \mathbin{@} u$ in Section 2. We will use the same abbreviation for operations which act in an extended context (such as $^\sim$, R, S, coe, coh).

Substitutions and terms are specified the same in the setoid model as in the graph model (Section 3). There is no need for R, S, T components because these are provable by proof irrelevance (unlike in the groupoid model [14, 16]). A substitution is a function between the $|-|$ sets which respects the $^\sim$ relations. In our syntactic reformulation we have a substitution between the $|-|$ and ^{01} components with a naturality property. Terms are interpreted by a term of the $|-|$ type in the $|-|$ context which respects the $^\sim$ relation.

In contrast with the graph translation, we cannot make $|\Gamma|$ the same as Γ. The reason is that $|(x : A) \to B|$ is not simply $(x : |A|) \to |B|$, but it also includes a proof that the function respects the $^\sim$ relations.

In the rest of this section, we specify the setoid translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}$ and then give the full implementation.

4.1 Specification

For a context, type, term and substitution, the translation gives the following. k stands for either 0 or 1, $(1-0)$ for 1 and $(1-1)$ for 0.

$$\frac{\vdash \Gamma}{\begin{array}{l}\vdash |\Gamma| \\ \vdash \Gamma^{01} \\ k_\Gamma \quad : \Gamma^{01} \Rightarrow |\Gamma| \\ \mathsf{R}_\Gamma \quad : |\Gamma| \Rightarrow \Gamma^{01} \\ k\mathsf{R}_\Gamma : k_\Gamma \circ \mathsf{R}_\Gamma = \mathsf{id}_{|\Gamma|} \\ \mathsf{S}_\Gamma \quad : \Gamma^{01} \Rightarrow \Gamma^{01} \\ k\mathsf{S}_\Gamma : k_\Gamma \circ \mathsf{S}_\Gamma = (1-k)_\Gamma\end{array}} \qquad \frac{\begin{array}{l}\vdash \Gamma \\ \rho_{01}\,\rho_{12} : \Theta \Rightarrow \Gamma^{01} \\ 1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12}\end{array}}{\begin{array}{l}\mathsf{T}_\Gamma\,\rho_0\,\rho_1 : \Theta \Rightarrow \Gamma^{01} \\ 0\mathsf{T}_\Gamma : 0_\Gamma \circ \mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12} = 0_\Gamma \circ \rho_{01} \\ 1\mathsf{T}_\Gamma : 1_\Gamma \circ \mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12} = 1_\Gamma \circ \rho_{12}\end{array}}$$

$$\frac{\begin{array}{l}\vdash \Gamma \\ \rho\,\rho' : \Theta \Rightarrow \Gamma^{01} \\ \forall k.k_\Gamma \circ \rho = k_\Gamma \circ \rho'\end{array}}{\mathsf{irr}_\Gamma : \rho = \rho'} \qquad \frac{\sigma : \Gamma \Rightarrow \Delta}{\begin{array}{ll}|\sigma| & : |\Gamma| \Rightarrow |\Delta| \\ \sigma^{01} & : \Gamma^{01} \Rightarrow \Delta^{01} \\ \mathsf{nat}k_\sigma & : k_\Delta \circ \sigma^{01} = |\sigma| \circ k_\Gamma\end{array}}$$

$$\frac{\Gamma \vdash_i A}{\begin{array}{l}|\Gamma| \vdash_i |A| \\ \Gamma^{01}, x_0 : |A|[0_\Gamma], x_1 : |A|[1_\Gamma] \vdash A^\sim\,x_0\,x_1 : \mathsf{Prop}_i \\ |\Gamma|, x : |A| \vdash \mathsf{R}_A\,x : \underline{A^\sim[\mathsf{R}_\Gamma]\,x\,x} \\ (\Gamma, x : A)^{01} \vdash \mathsf{S}_A\,x_0\,x_1\,x_{01} : \underline{A^\sim[\mathsf{S}_\Gamma]\,x_1\,x_0} \\ \Gamma^{01}, x_0 : |A|[0_\Gamma] \vdash \mathsf{coe}_A\,x_0 : \underline{|A|[1_\Gamma]} \\ \Gamma^{01}, x_0 : |A|[0_\Gamma] \vdash \mathsf{coh}_A\,x_0 : \underline{A^\sim\,x_0\,(\mathsf{coe}_A\,x_0)} \\ \mathsf{coeR}_A : \mathsf{coe}_A[\mathsf{R}_\Gamma]\,x = x\end{array}}$$

$$\frac{\begin{array}{l}\Gamma \vdash_i A \\ \rho_{01}\,\rho_{12} : \Theta \Rightarrow \Gamma^{01} \\ 1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12} \\ \Theta \vdash t_0 : |A|[0_\Gamma \circ \rho_{01}] \\ \Theta \vdash t_1 : |A|[1_\Gamma \circ \rho_{01}] \\ \Theta \vdash t_2 : |A|[1_\Gamma \circ \rho_{12}] \\ \Theta \vdash t_{01} : \underline{A^\sim[\rho_{01}]\,t_0\,t_1} \\ \Theta \vdash t_{12} : \underline{A^\sim[\rho_{12}]\,t_1\,t_2}\end{array}}{\Theta \vdash \mathsf{T}_A\,\rho_{01}\,\rho_{12}\,t_0\,t_1\,t_2\,t_{01}\,t_{12} : \underline{A^\sim[\mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12}]\,t_0\,t_2}}$$

$$\frac{\Gamma \vdash t : A}{\begin{array}{l}|\Gamma| \vdash |t| : |A| \\ \Gamma^{01} \vdash t^\sim : \underline{A^\sim\,(|t|[0_\Gamma])\,(|t|[1_\Gamma])}\end{array}}$$

Now we finished specifying the setoid translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}$. Now we list some useful direct consequences of the specification that we will use in the implementation later.

Prop-irrelevance and irr_Γ have the following consequences. Given a $\sigma : \Gamma \Rightarrow \Delta$, we have $\mathsf{R}_\Delta \circ |\sigma| = \sigma^{01} \circ \mathsf{R}_\Gamma$ by $0\mathsf{R}_\Delta$, $1\mathsf{R}_\Delta$, $\mathsf{nat0}_\sigma$, $\mathsf{nat1}_\sigma$, $0\mathsf{R}_\Gamma$, $1\mathsf{R}_\Gamma$ and irr_Δ. Similarly, we have $\mathsf{S}_\Delta \circ \sigma^{01} = \sigma^{01} \circ \mathsf{S}_\Gamma$, $\mathsf{S}_\Gamma \circ \mathsf{S}_\Gamma = \mathsf{id}_{\Gamma^{01}}$, $\mathsf{S}_\Gamma \circ \mathsf{R}_\Gamma = \mathsf{R}_\Gamma$. For all $\Gamma \vdash t : A$ we have $t^\sim[\mathsf{R}_\Gamma] = \mathsf{R}_A\, t$ and $\mathsf{coh}_A[\mathsf{R}_\Gamma]\, t = \mathsf{R}_A\, t$.

Coercion and coherence in the other direction are defined using symmetry as follows.

$$\begin{aligned}
&\Gamma^{01}, x_1 : |A|[1_\Gamma] \vdash \mathsf{coe}^*_A\, x_1 := \mathsf{coe}_A[\mathsf{S}_\Gamma]\, x_1 && : |A|[0_\Gamma]\\
&\Gamma^{01}, x_1 : |A|[1_\Gamma] \vdash \mathsf{coh}^*_A\, x_1 := \mathsf{S}_A[\mathsf{S}_\Gamma]\, x_1\, (\mathsf{coe}^*_A\, x_1)\, (\mathsf{coh}_A[\mathsf{S}_\Gamma]\, x_1) && : \underline{A^\sim\, (\mathsf{coe}^*_A\, x_1)\, x_1}
\end{aligned}$$

From coeR_A we have that $\mathsf{coe}^*_A[\mathsf{R}_\Gamma]\, x = x$ and $\mathsf{coh}^*_A[\mathsf{R}_\Gamma]\, x = \mathsf{R}_A\, x$.

We define transitivity for three subsitutions as follows.

$$\frac{\rho_{01}\, \rho_{12}\, \rho_{23} : \Theta \Rightarrow \Gamma^{01} \qquad 1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12} \qquad 1_\Gamma \circ \rho_{12} = 0_\Gamma \circ \rho_{23}}{{\mathsf{T}^3}_\Gamma\, \rho_{01}\, \rho_{12}\, \rho_{23} := \mathsf{T}_\Gamma\, \rho_{01}\, (\mathsf{T}_\Gamma\, \rho_{12}\, \rho_{23}) : \Theta \Rightarrow \Gamma^{01}}$$

Similarly for three term equalities.

$$\begin{array}{l}
\rho_{01}\, \rho_{12}\, \rho_{23} : \Theta \Rightarrow \Gamma^{01}\\
1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12}\\
1_\Gamma \circ \rho_{12} = 0_\Gamma \circ \rho_{23}\\
\Theta \vdash t_0 : |A|[0_\Gamma \circ \rho_{01}]\\
\Theta \vdash t_1 : |A|[1_\Gamma \circ \rho_{01}]\\
\Theta \vdash t_2 : |A|[1_\Gamma \circ \rho_{12}]\\
\Theta \vdash t_3 : |A|[1_\Gamma \circ \rho_{23}]\\
\Theta \vdash t_{01} : \underline{A^\sim[\rho_{01}]\, t_0\, t_1}\\
\Theta \vdash t_{12} : \underline{A^\sim[\rho_{12}]\, t_1\, t_2}\\
\Theta \vdash t_{23} : \underline{A^\sim[\rho_{23}]\, t_2\, t_3}
\end{array}$$

${\mathsf{T}^3}_A\, \rho_{01}\, \rho_{12}\, \rho_{23}\, t_0\, t_1\, t_2\, t_3\, t_{01}\, t_{12}\, t_{23} :=$
$\mathsf{T}_A\, \rho_{01}\, (\mathsf{T}_\Gamma\, \rho_{12}\, \rho_{23})\, t_0\, t_1\, t_3\, t_{01}\, (\mathsf{T}_A\, \rho_{12}\, \rho_{23}\, t_1\, t_2\, t_3\, t_{12}\, t_{23}) : \underline{A^\sim[{\mathsf{T}^3}_\Gamma\, \rho_{01}\, \rho_{12}\, \rho_{23}]\, t_0\, t_3}$

4.2 Implementation

We list the implementation of the operations $|-|$, $-^\sim$, ..., coeR for contexts, types, substitutions and terms. Equality proofs are given by equational reasoning, we write the the proofs of equalities above the $=$ symbols, when they have an explicit name.

Contexts:

$$\begin{aligned}
|\cdot| &:= \cdot\\
|\Gamma, x : A| &:= |\Gamma|, x : |A|\\
{}^{\cdot 01} &:= \cdot\\
(\Gamma, x : A)^{01} &:= \Gamma^{01}, x_0 : |A|[0_\Gamma], x_1 : |A|[1_\Gamma], x_{01} : \underline{A^\sim\, x_0\, x_1}\\
k_{\cdot} &:= \epsilon\\
k_{\Gamma, x:A} &:= (k_\Gamma,\, x \mapsto x_k)\\
\mathsf{R}_{\cdot} &:= \epsilon\\
\mathsf{R}_{\Gamma, x:A} &:= (\mathsf{R}_\Gamma,\, x_0 \mapsto x,\, x_1 \mapsto x,\, x_{01} \mapsto \mathsf{R}_A\, x)\\
k\mathsf{R}_{\cdot} &: \quad k_{\cdot} \circ \mathsf{R}_{\cdot} = \epsilon = \mathsf{id}_{|\cdot|}
\end{aligned}$$

$$
\begin{array}{lcl}
k\mathsf{R}_{\Gamma,x:A} & : & k_{\Gamma,x:A} \circ \mathsf{R}_{\Gamma,x:A} = (k_\Gamma \circ \mathsf{R}_\Gamma,\, x \mapsto x) \overset{k\mathsf{R}_\Gamma}{=} (\mathsf{id}_{|\Gamma|},\, x \mapsto x) = \mathsf{id}_{|\Gamma,x:A|} \\
\mathsf{S}. & := & \epsilon \\
\mathsf{S}_{\Gamma,x:A} & := & (\mathsf{S}_\Gamma,\, x_0 \mapsto x_1,\, x_1 \mapsto x_0,\, x_{01} \mapsto \mathsf{S}_A\, x_0\, x_1\, x_{01}) \\
k\mathsf{S}. & : & k. \circ \mathsf{S}. = \epsilon = (1-k). \\
k\mathsf{S}_{\Gamma,x:A} & : & k_{\Gamma,x:A} \circ \mathsf{S}_{\Gamma,x:A} = (k_\Gamma \circ \mathsf{S}_\Gamma,\, x \mapsto x_{1-k}) \overset{k\mathsf{S}_\Gamma}{=} ((1-k)_\Gamma,\, x \mapsto x_{1-k}) = \\
 & & (1-k)_{\Gamma,x:A} \\
\mathsf{T}.\, \rho\, \rho' & := & \epsilon \\
\mathsf{T}_{\Gamma,x:A}\, (\rho_{01},\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t_{01})\, (\rho_{12},\, x_0 \mapsto t_1,\, x_1 \mapsto t_2,\, x_{01} \mapsto t_{12}) := & & \\
 & & (\mathsf{T}_\Gamma\, \rho_{01}\, \rho_{12},\, x_0 \mapsto t_0,\, x_1 \mapsto t_2,\, x_{01} \mapsto \mathsf{T}_A\, \rho_{01}\, \rho_{12}\, t_0\, t_1\, t_2\, t_{01}\, t_{12}) \\
k\mathsf{T}. & : & k. \circ \mathsf{T}.\, \rho_0\, \rho_1 = \epsilon = k. \circ \rho_k \\
0\mathsf{T}_{\Gamma,x:A} & : & 0_{\Gamma,x:A} \circ \mathsf{T}_{\Gamma,x:A}\, (\rho_{01},\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t_{01}) \\
 & & \qquad (\rho_{12},\, x_0 \mapsto t_1,\, x_1 \mapsto t_2,\, x_{01} \mapsto t_{12}) = \\
 & & (0_\Gamma \circ \mathsf{T}_\Gamma\, \rho_{01}\, \rho_{12}, x \mapsto t_0) \overset{0\mathsf{T}_\Gamma}{=} \\
 & & 0_{\Gamma,x:A} \circ (\rho_{01},\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t_{01}) \\
1\mathsf{T}_{\Gamma,x:A} & : & 1_{\Gamma,x:A} \circ \mathsf{T}_{\Gamma,x:A}\, (\rho_{01},\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t_{01}) \\
 & & \qquad (\rho_{12},\, x_0 \mapsto t_1,\, x_1 \mapsto t_2,\, x_{01} \mapsto t_{12}) = \\
 & & (1_\Gamma \circ \mathsf{T}_\Gamma\, \rho_{01}\, \rho_{12}, x \mapsto t_2) \overset{1\mathsf{T}_\Gamma}{=} \\
 & & 1_{\Gamma,x:A} \circ (\rho_{12},\, x_0 \mapsto t_1,\, x_1 \mapsto t_2,\, x_{01} \mapsto t_{12}) \\
\mathsf{irr}. & : & \rho = \epsilon = \rho' \\
\mathsf{irr}_{\Gamma,x:A} & : & (\rho,\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t_{01}) \overset{\mathsf{irr}_{A^\sim[\rho]\, t_0\, t_1}}{=} \\
 & & (\rho,\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t'_{01}) \overset{\mathsf{irr}_\Gamma}{=} \\
 & & (\rho',\, x_0 \mapsto t_0,\, x_1 \mapsto t_1,\, x_{01} \mapsto t'_{01})
\end{array}
$$

Types:

$$
\begin{array}{lcl}
|A[\sigma]| & := & |A|[|\sigma|] \\
|(x:A) \to B| & := & (f : (x : |A|) \to |B|) \times \\
 & & \underline{(x_0\, x_1 : |A|)(x_{01} : \underline{A^\sim[\mathsf{R}_\Gamma]\, x_0\, x_1}) \to B^\sim[\mathsf{R}_\Gamma]\, (f \mathbin{@} x_0)\, (f \mathbin{@} x_1)} \\
|(x:A) \times B| & := & (x : |A|) \times |B| \\
|\mathsf{Bool}| & := & \mathsf{Bool} \\
|\mathsf{Prop}_i| & := & \mathsf{Prop}_i \\
|\underline{a}| & := & \underline{|a|}
\end{array}
$$

$$
\begin{array}{lcl}
(A[\sigma])^\sim\, x_0\, x_1 & := & A^\sim[\sigma^{01}]\, x_0\, x_1 \\
((x:A) \to B)^\sim\, f_0\, f_1 & := & (x_0 : |A|[0_\Gamma])(x_1 : |A|[1_\Gamma])(x_{01} : \underline{A^\sim\, x_0\, x_1}) \to \\
 & & \qquad B^\sim\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0)\, (\mathsf{pr}_0\, f_1 \mathbin{@} x_1) \\
((x:A) \times B)^\sim\, z_0\, z_1 & := & \mathsf{let}\, x_0 \mapsto \mathsf{pr}_0\, z_0,\, x_1 \mapsto \mathsf{pr}_0\, z_1\, \mathsf{in} \\
 & & \qquad (x_{01} : A^\sim\, x_0\, x_1) \times B^\sim\, (\mathsf{pr}_1\, z_0)\, (\mathsf{pr}_1\, z_1) \\
\mathsf{Bool}^\sim\, x_0\, x_1 & := & \mathsf{if}\, x_0\, \mathsf{then}\, (\mathsf{if}\, x_1\, \mathsf{then}\, \top\, \mathsf{else}\, \bot)\, \mathsf{else}\, (\mathsf{if}\, x_1\, \mathsf{then}\, \bot\, \mathsf{else}\, \top)
\end{array}
$$

$$
\begin{aligned}
&\mathsf{Prop}_i{}^{\sim}\, z_0\, z_1 && := (\underline{z_0} \to z_1) \times (\underline{z_1} \to z_0)\\
&\underline{a}^{\sim}\, x_0\, x_1 && := \top
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{R}_{A[\sigma]}\, x && := \mathsf{R}_A[|\sigma|]\, x\\
&\mathsf{R}_{(x:A)\to B}\, z && := \mathsf{pr}_1\, z\\
&\mathsf{R}_{(x:A)\times B}\, z && := (\mathsf{R}_A\, (\mathsf{pr}_0\, z), \mathsf{R}_B[x \mapsto \mathsf{pr}_0\, z]\, (\mathsf{pr}_1\, z))\\
&\mathsf{R}_{\mathsf{Bool}}\, x && := \mathsf{if}\ x\ \mathsf{then}\ \mathsf{tt}\ \mathsf{else}\ \mathsf{tt}\\
&\mathsf{R}_{\mathsf{Prop}_i}\, z && := (\lambda x.x, \lambda x.x)\\
&\mathsf{R}_{\underline{a}}\, x && := \mathsf{tt}
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{S}_{A[\sigma]}\, x_0\, x_1\, x_{01} && := \mathsf{S}_A[\sigma^{01}]\, x_0\, x_1\, x_{01}\\
&\mathsf{S}_{(x:A)\to B}\, f_0\, f_1\, f_{01} :=\\
&\quad \lambda x_0\, x_1\, x_{01}.\mathsf{S}_B[x_0 \mapsto x_1,\, x_1 \mapsto x_0,\, x_{01} \mapsto \mathsf{S}_A[\mathsf{S}_\Gamma]\, x_0\, x_1\, x_{01}]\\
&\quad (\mathsf{pr}_0\, f_0 \mathbin{@} x_1)\, (\mathsf{pr}_0\, f_1 \mathbin{@} x_0)\, (f_{01} \mathbin{@} x_1 \mathbin{@} x_0 \mathbin{@} \mathsf{S}_A[\mathsf{S}_\Gamma]\, x_0\, x_1\, x_{01})\\
&\mathsf{S}_{(x:A)\times B}\, z_0\, z_1\, z_{01} && := \mathsf{let}\, x_0 \mapsto \mathsf{pr}_0\, z_0,\, x_1 \mapsto \mathsf{pr}_0\, z_1,\, x_{01} \mapsto \mathsf{pr}_0\, z_{01}\, \mathsf{in}\\
&&& \quad (\mathsf{S}_A\, x_0\, x_1\, x_{01}, \mathsf{S}_B\, (\mathsf{pr}_1\, z_0)\, (\mathsf{pr}_1\, z_1)\, (\mathsf{pr}_1\, z_{01}))\\
&\mathsf{S}_{\mathsf{Bool}}\, x_0\, x_1\, x_{01} && :=\\
&\quad \mathsf{if}\ x_1\ \mathsf{then}\ (\mathsf{if}\ x_0\ \mathsf{then}\ \mathsf{tt}\ \mathsf{else}\ \mathsf{abort}\ x_{01})\ \mathsf{else}\ (\mathsf{if}\ x_0\ \mathsf{then}\ \mathsf{abort}\ x_{01}\ \mathsf{else}\ \mathsf{tt})\\
&\mathsf{S}_{\mathsf{Prop}_i}\, z_0\, z_1\, z_{01} && := (\mathsf{pr}_1\, z_{01}, \mathsf{pr}_0\, z_{01})\\
&\mathsf{S}_{\underline{a}}\, x_0\, x_1\, x_{01} && := \mathsf{tt}
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{coe}_{A[\sigma]}\, x_0 && := \mathsf{coe}_A[\sigma^{01}]\, x_0\\
&\mathsf{coe}_{(x:A)\to B}\, f_0 := (\lambda x_1.\mathsf{coe}_B[x_0 \mapsto \mathsf{coe}^*_A\, x_1, x_{01} \mapsto \mathsf{coh}^*_A\, x_1]\, (\mathsf{pr}_0\, f_0 \mathbin{@} \mathsf{coe}^*_A\, x_1),\\
&\quad \lambda x_0\, x_1\, x_{01}.\mathsf{let}\\
&\quad x_0 \mapsto x_0,\\
&\quad x_1 \mapsto \mathsf{coe}^*_A\, x_0,\\
&\quad x_2 \mapsto \mathsf{coe}^*_A\, x_1,\\
&\quad x_3 \mapsto x_1,\\
&\quad x_{01} \mapsto \mathsf{coh}_A[\mathsf{S}_\Gamma]\, x_0,\\
&\quad x_{12} \mapsto \mathsf{T}^3{}_A\, \mathsf{id}_{\Gamma^{01}}\, (\mathsf{R}_\Gamma \circ 1_\Gamma)\, \mathsf{S}_\Gamma\, (\mathsf{coe}^*_A\, x_0)\, x_0\, x_1\, (\mathsf{coe}^*_A\, x_1)\\
&\qquad\qquad (\mathsf{coh}^*_A\, x_0)\, x_{01}\, (\mathsf{coh}_A[\mathsf{S}_\Gamma]\, x_1)\\
&\quad x_{23} \mapsto \mathsf{coh}^*_A\, x_1\ \mathsf{in}\ \mathsf{let}\\
&\quad y_0 \mapsto (\mathsf{coe}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0))[x_0 \mapsto x_1,\, x_1 \mapsto x_0,\, x_{01} \mapsto \mathsf{coh}^*_A\, x_0],\\
&\quad y_1 \mapsto \mathsf{pr}_0\, f_0 \mathbin{@} x_1,\\
&\quad y_2 \mapsto \mathsf{pr}_0\, f_0 \mathbin{@} x_2,\\
&\quad y_3 \mapsto (\mathsf{coe}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0))[x_0 \mapsto x_2,\, x_1 \mapsto x_3,\, x_{01} \mapsto x_{23}],\\
&\quad y_{01} \mapsto (\mathsf{S}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0)\, (\mathsf{coe}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0))\, (\mathsf{coh}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0)))[x_0 \mapsto x_1,\\
&\qquad\quad x_1 \mapsto x_0,\, x_{01} \mapsto \mathsf{coh}^*_A\, x_0],\\
&\quad y_{12} \mapsto \mathsf{pr}_1\, f_0 \mathbin{@} x_1 \mathbin{@} x_2 \mathbin{@} x_{12},\\
&\quad y_{23} \mapsto (\mathsf{coh}_B\, (\mathsf{pr}_0\, f_0 \mathbin{@} x_0))[x_0 \mapsto x_2,\, x_1 \mapsto x_3,\, x_{01} \mapsto x_{23}]\ \mathsf{in}
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{T}^3{}_B\,\mathsf{S}_\Gamma\,(\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_0 \mapsto x_1,\, x_1 \mapsto x_2,\, x_{01} \mapsto x_{12})\\
&\quad(\mathsf{id}_{\Gamma^{01}},\, x_0 \mapsto x_2,\, x_1 \mapsto x_3,\, x_{01} \mapsto x_{23})\, y_0\, y_1\, y_2\, y_3\, y_{01}\, y_{12}\, y_{23})\\
\mathsf{coe}_{(x:A)\times B}\, z_0 &:= \mathsf{let}\, x_0 \mapsto \mathsf{pr}_0\, z_0,\, x_1 \mapsto \mathsf{coe}_A\,(\mathsf{pr}_0\, z_0),\, x_{01} \mapsto \mathsf{coh}_A\,(\mathsf{pr}_0\, z_0)\,\mathsf{in}\\
&\qquad(\mathsf{coe}_A\, x_0, \mathsf{coe}_B\,(\mathsf{pr}_1\, z_0))\\
\mathsf{coe}_{\mathsf{Bool}}\, x_0 &:= x_0\\
\mathsf{coe}_{\mathsf{Prop}_i}\, z_0 &:= z_0\\
\mathsf{coe}_{\underline{a}}\, x_0 &:= \mathsf{pr}_0\, a^\sim \mathbin{@} x_0
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{coh}_{A[\sigma]}\, x_0 &:= \mathsf{coh}_A[\sigma^{01}]\, x_0\\
\mathsf{coh}_{(x:A)\to B}\, f_0 &:= \lambda x_0\, x_1\, x_{01}.\mathsf{let}\, x_2 \mapsto \mathsf{coe}^*_A\, x_1,\, x_{12} \mapsto \mathsf{coh}_A[\mathsf{S}_\Gamma]\, x_1\,\mathsf{in}\\
&\quad\mathsf{let}\, x_{02} \mapsto \mathsf{T}_A\,\mathsf{id}_{\Gamma^{01}}\,\mathsf{S}_\Gamma\, x_0\, x_1\, x_2\, x_{01}\, x_{12},\, x_{21} \mapsto \mathsf{coh}^*_A\, x_1\,\mathsf{in}\\
&\quad\mathsf{T}_B\,(\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_1 \mapsto x_2,\, x_{01} \mapsto x_{02})\,(x_0 \mapsto x_2, x_{01} \mapsto x_{21})\\
&\qquad(\mathsf{pr}_0\, f_0 \mathbin{@} x_0)\,(\mathsf{pr}_0\, f_0 \mathbin{@} x_2)\,(\mathsf{coe}_B[x_0 \mapsto x_2,\, x_{01} \mapsto x_{21}]\,(\mathsf{pr}_0\, f_0 \mathbin{@} x_2))\\
&\qquad(\mathsf{pr}_1\, f_0 \mathbin{@} x_0 \mathbin{@} x_2 \mathbin{@} x_{02})\,(\mathsf{coh}_B[x_0 \mapsto x_2, x_{01} \mapsto x_{21}]\,(\mathsf{pr}_0\, f_0 \mathbin{@} x_2))\\
\mathsf{coh}_{(x:A)\times B}\, z_0 &:= \mathsf{let}\, x_0 \mapsto \mathsf{pr}_0\, z_0,\, x_1 \mapsto \mathsf{coe}_A\,(\mathsf{pr}_0\, z_0),\, x_{01} \mapsto \mathsf{coh}_A\,(\mathsf{pr}_0\, z_0)\,\mathsf{in}\\
&\qquad(\mathsf{coh}_A\, x_0, \mathsf{coh}_B\,(\mathsf{pr}_1\, z_0))\\
\mathsf{coh}_{\mathsf{Bool}}\, x_0 &:= \mathsf{if}\, x_0\,\mathsf{then}\,\mathsf{tt}\,\mathsf{else}\,\mathsf{tt}\\
\mathsf{coh}_{\mathsf{Prop}_i}\, z_0 &:= (\lambda x.x, \lambda x.x)\\
\mathsf{coh}_{\underline{a}}\, x_0 &:= \mathsf{tt}
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{coeR}_{A[\sigma]} &: \mathsf{coe}_{A[\sigma]}[\mathsf{R}_\Gamma]\, x_0 = \mathsf{coe}_A[\sigma^{01} \circ \mathsf{R}_\Gamma]\, x_0 = \mathsf{coe}_A[\mathsf{R}_\Delta \circ |\sigma|]\, x_0 =\\
&\qquad(\mathsf{coe}_A[\mathsf{R}_\Delta]\, x_0)[|\sigma|] \overset{\mathsf{coeR}_A}{=} x_0\\
\mathsf{coeR}_{(x:A)\to B} &: \mathsf{coe}_{(x:A)\to B}[\mathsf{R}_\Gamma]\, f_0 =\\
&\quad(\lambda x_1.\mathsf{coe}_B\big[\mathsf{R}_\Gamma, x_0 \mapsto \mathsf{coe}^*_A[\mathsf{R}_\Gamma]\, x_1, x_{01} \mapsto \mathsf{coh}^*_A[\mathsf{R}_\Gamma]\, x_1\big]\,(\mathsf{pr}_0\, f_0 \mathbin{@} \mathsf{coe}^*_A[\mathsf{R}_\Gamma]\, x_1), \dots) \overset{\mathsf{coeR}_A}{=}\\
&\quad(\lambda x.\mathsf{coe}_B[\mathsf{R}_{\Gamma,x:A}]\,(\mathsf{pr}_0\, f_0 \mathbin{@} x), \dots) \overset{\mathsf{coeR}_B}{=} (\lambda x.(\mathsf{pr}_0\, f_0 \mathbin{@} x), \dots) = (\mathsf{pr}_0\, f_0, \mathsf{pr}_1\, f_0) = f_0\\
\mathsf{coeR}_{(x:A)\times B} &: \mathsf{coe}_{(x:A)\times B}[\mathsf{R}_\Gamma]\, z_0 =\\
&\quad\mathsf{let}\, x_0 \mapsto \mathsf{pr}_0\, z_0,\, x_1 \mapsto \mathsf{coe}_A[\mathsf{R}_\Gamma]\,(\mathsf{pr}_0\, z_0),\, x_{01} \mapsto \mathsf{coh}_A[\mathsf{R}_\Gamma]\,(\mathsf{pr}_0\, z_0)\,\mathsf{in}\\
&\quad(\mathsf{coe}_A[\mathsf{R}_\Gamma]\, x_0, \mathsf{coe}_B[\mathsf{R}_\Gamma]\,(\mathsf{pr}_1\, z_0)) \overset{\mathsf{coeR}_A}{=}\\
&\quad\mathsf{let}\, x \mapsto \mathsf{pr}_0\, z_0\,\mathsf{in}\,(x, \mathsf{coe}_B[\mathsf{R}_{\Gamma,x:A}]\,(\mathsf{pr}_1\, z_0)) \overset{\mathsf{coeR}_B}{=} (\mathsf{pr}_0\, z_0, \mathsf{pr}_1\, z_0) = z_0\\
\mathsf{coeR}_{\mathsf{Bool}} &: \mathsf{coe}_{\mathsf{Bool}}[\mathsf{R}_\Gamma]\, x_0 = x_0\\
\mathsf{coeR}_{\mathsf{Prop}_i} &: \mathsf{coe}_{\mathsf{Prop}_i}[\mathsf{R}_\Gamma]\, z_0 = z_0\\
\mathsf{coeR}_{\underline{a}}\, x_0 &: \mathsf{coe}_{\underline{a}}[\mathsf{R}_\Gamma]\, x_0 = \mathsf{pr}_0\,(a^\sim[\mathsf{R}_\Gamma]) \mathbin{@} x_0 = \mathsf{pr}_0\,(\mathsf{R}_{\mathsf{Prop}_i}\, a) \mathbin{@} x_0 =\\
&\qquad(\lambda x.x) \mathbin{@} x_0 = x_0
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{T}_{A[\sigma]}\, \rho_{01}\, \rho_{12}\; t_0\, t_1\, t_2\, t_{01}\, t_{12} &:= \mathsf{T}_A\,(\sigma^{01} \circ \rho_{01})\,(\sigma^{01} \circ \rho_{12})\, t_0\, t_1\, t_2\, t_{01}\, t_{12}\\
\mathsf{T}_{(x:A)\to B}\, \rho_{01}\, \rho_{12}\, t_0\, t_1\, t_2\, t_{01}\, t_{12} &:=\\
&\quad\lambda x_0\, x_2\, x_{02}.\mathsf{let}\, x_1 \mapsto \mathsf{coe}_A[\rho_{01}]\, x_0,\, x_{01} \mapsto \mathsf{coh}_A[\rho_{01}]\, x_0\,\mathsf{in}\\
&\qquad\mathsf{let}\, x_{12} \mapsto \mathsf{T}_A\,(\mathsf{S}_\Gamma \circ \rho_{01})\,(\mathsf{T}_\Gamma\, \rho_{01}\, \rho_{12})\, x_1\, x_0\, x_2\,(\mathsf{S}_A[\rho_{01}]\, x_0\, x_1\, x_{01})\, x_{02}\,\mathsf{in}
\end{aligned}
$$

$$\mathsf{T}_B\,\rho_{01}\,(\rho_{12},\,x_0\mapsto x_1,\,x_1\mapsto x_2,\,x_{01}\mapsto x_{12})\,(\mathsf{pr}_0\,t_0\mathbin{@}x_0)\,(\mathsf{pr}_0\,t_1\mathbin{@}x_1)\,(\mathsf{pr}_0\,t_2\mathbin{@}x_2)$$
$$(t_{01}\mathbin{@}x_0\mathbin{@}x_1\mathbin{@}x_{01})\,(t_{12}\mathbin{@}x_1\mathbin{@}x_2\mathbin{@}x_{12})$$

$$\mathsf{T}_{(x:A)\times B}\,\rho_{01}\,\rho_{12}\,t_0\,t_1\,t_2\,t_{01}\,t_{12}\;:=$$
$$\mathsf{let}\,x_0\mapsto\mathsf{pr}_0\,t_0,\,x_1\mapsto\mathsf{pr}_0\,t_1,\,x_2\mapsto\mathsf{pr}_0\,t_2,\,x_{01}\mapsto\mathsf{pr}_0\,t_{01},\,x_{12}\mapsto\mathsf{pr}_0\,t_{12}\,\mathsf{in}$$
$$(\mathsf{T}_A\,\rho_{01}\,\rho_{12}\,x_0\,x_1\,x_2\,x_{01}\,x_{12},$$
$$\mathsf{T}_B\,\rho_{01}\,(\rho_{12},\,x_0\mapsto x_1,\,x_1\mapsto x_2,\,x_{01}\mapsto x_{12})\,(\mathsf{pr}_1\,t_0)\,(\mathsf{pr}_1\,t_1)\,(\mathsf{pr}_1\,t_2)\,(\mathsf{pr}_1\,t_{01})\,(\mathsf{pr}_1\,t_{12}))$$

$$\mathsf{T}_{\mathsf{Bool}}\,\rho_{01}\,\rho_{12}\,x_0\,x_1\,x_2\,x_{01}\,x_{12}\;:=$$
$$\mathsf{if}\,x_0\,\mathsf{then}\,(\mathsf{if}\,x_1\,\mathsf{then}\,(\mathsf{if}\,x_2\,\mathsf{then}\,\mathsf{tt}\,\mathsf{else}\,\mathsf{abort}\,x_{12})\,\mathsf{else}\,\mathsf{abort}\,x_{01})\,\mathsf{else}$$
$$(\mathsf{if}\,x_1\,\mathsf{then}\,\mathsf{abort}\,x_{01}\,\mathsf{else}\,(\mathsf{if}\,x_2\,\mathsf{abort}\,x_{12}\,\mathsf{else}\,\mathsf{tt}))$$

$$\mathsf{T}_{\mathsf{Prop}_i}\,\rho_{01}\,\rho_{12}\,z_0\,z_1\,z_2\,z_{01}\,z_{12}\;:=$$
$$(\lambda x_0.\mathsf{pr}_0\,z_{12}\mathbin{@}(\mathsf{pr}_0\,z_{01}\mathbin{@}x_0),\lambda x_2.\mathsf{pr}_1\,z_{12}\mathbin{@}(\mathsf{pr}_1\,z_{01}\mathbin{@}x_2))$$

$$\mathsf{T}_{\underline{a}}\,\rho_{01}\,\rho_{12}\,x_0\,x_1\,x_2\,x_{01}\,x_{12}\;:=\mathsf{tt}$$

Substitutions:

$$|\epsilon|\;:=\epsilon$$
$$|\sigma,\,x\mapsto t|\;:=(|\sigma|,\,x\mapsto|t|)$$
$$|\mathsf{id}_\Gamma|\;:=\mathsf{id}_{|\Gamma|}$$
$$|\sigma\circ\delta|\;:=|\sigma|\circ|\delta|$$
$$\epsilon^{01}\;:=\epsilon$$
$$(\sigma,\,x\mapsto t)^{01}\;:=(\sigma^{01},\,x_0\mapsto|t|[0_\Gamma],\,x_1\mapsto|t|[1_\Gamma],\,x_{01}\mapsto t^\sim)$$
$$\mathsf{id}_\Gamma{}^{01}\;:=\mathsf{id}_{\Gamma^{01}}$$
$$(\sigma\circ\delta)^{01}\;:=\sigma^{01}\circ\delta^{01}$$
$$\mathsf{nat}k_\epsilon\;:\;k_.\circ\epsilon^{01}=\epsilon=|\epsilon|\circ k_\Gamma$$
$$\mathsf{nat}k_{(\sigma,\,x\mapsto t)}\;:\;k_{\Delta,x:A}\circ(\sigma,\,x\mapsto t)^{01}=(k_\Delta\circ\sigma^{01},\,x\mapsto|t|[k_\Gamma])\overset{\mathsf{nat}k_\sigma}{=}$$
$$(|\sigma|\circ k_\Gamma,\,x\mapsto|t|[k_\Gamma])=|\sigma,\,x\mapsto t|\circ k_\Gamma$$
$$\mathsf{nat}k_{\mathsf{id}_\Gamma}\;:\;k_\Gamma\circ\mathsf{id}_\Gamma{}^{01}=k_\Gamma\circ\mathsf{id}_{\Gamma^{01}}=k_\Gamma=\mathsf{id}_{|\Gamma|}\circ k_\Gamma=|\mathsf{id}_\Gamma|\circ k_\Gamma$$
$$\mathsf{nat}k_{\sigma\circ\delta}\;:\;k_\Delta\circ\sigma\circ\delta^{01}=k_\Delta\circ\sigma^{01}\circ\delta^{01}\overset{\mathsf{nat}k_\sigma}{=}|\sigma|\circ k_\Theta\circ\delta^{01}\overset{\mathsf{nat}k_\delta}{=}$$
$$|\sigma|\circ|\delta|\circ k_\Gamma=|\sigma\circ\delta|\circ k_\Gamma$$

Terms:

$$|x|\;:=x$$
$$|t[\sigma]|\;:=|t|[|\sigma|]$$
$$|\lambda x.t|\;:=(\lambda x.|t|,\lambda x_0\,x_1\,x_{01}.t^\sim[\mathsf{R}_\Gamma])$$
$$|t\mathbin{@}x|\;:=\mathsf{pr}_0\,|t|\mathbin{@}x$$
$$|\Pi\beta|\;:\;|(\lambda x.t)\mathbin{@}x|=\mathsf{pr}_0\,|\lambda x.t|\mathbin{@}x=(\lambda x.|t|)\mathbin{@}x=|t|$$
$$|\Pi\eta|\;:\;|\lambda x.t\mathbin{@}x|=(\lambda x.|t\mathbin{@}x|,\lambda x_0\,x_1\,x_{01}.(t\mathbin{@}x)^\sim)=$$
$$(\lambda x.\mathsf{pr}_0\,|t|\mathbin{@}x,\lambda x_0\,x_1\,x_{01}.t^\sim[\mathsf{R}_\Gamma]\,x_0\,x_1\,x_{01})=(\mathsf{pr}_0\,|t|,t^\sim[\mathsf{R}_\Gamma])=$$
$$(\mathsf{pr}_0\,|t|,\mathsf{R}_{(x:A)\to B}\,t)=(\mathsf{pr}_0\,|t|,\mathsf{pr}_1\,|t|)=|t|$$
$$|(u,v)|\;:=(|u|,|v|)$$

$$
\begin{array}{lll}
|\mathsf{pr}_0\, t| & := & \mathsf{pr}_0\, |t| \\
|\mathsf{pr}_1\, t| & := & \mathsf{pr}_1\, |t| \\
|\Sigma\beta_0| & : & |\mathsf{pr}_0\, (u, v)| = \mathsf{pr}_0\, |(u, v)| = \mathsf{pr}_0\, (|u|, |v|) = |u| \\
|\Sigma\beta_1| & : & |\mathsf{pr}_1\, (u, v)| = \mathsf{pr}_1\, |(u, v)| = \mathsf{pr}_1\, (|u|, |v|) = |v| \\
|\Sigma\eta| & : & |(\mathsf{pr}_0\, t, \mathsf{pr}_1\, t)| = (|\mathsf{pr}_0\, t|, |\mathsf{pr}_1\, t|) = (\mathsf{pr}_0\, |t|, \mathsf{pr}_1\, |t|) = |t| \\
|\mathsf{true}| & := & \mathsf{true} \\
|\mathsf{false}| & := & \mathsf{false} \\
|\mathsf{if}\, t\, \mathsf{then}\, u\, \mathsf{else}\, v| & := & \mathsf{if}\, |t|\, \mathsf{then}\, |u|\, \mathsf{else}\, |v| \\
|\mathsf{Bool}\beta\mathsf{true}| & : & |\mathsf{if}\, \mathsf{true}\, \mathsf{then}\, u\, \mathsf{else}\, v| = \mathsf{if}\, \mathsf{true}\, \mathsf{then}\, |u|\, \mathsf{else}\, |v| = |u| \\
|\mathsf{Bool}\beta\mathsf{false}| & : & |\mathsf{if}\, \mathsf{false}\, \mathsf{then}\, u\, \mathsf{else}\, v| = \mathsf{if}\, \mathsf{false}\, \mathsf{then}\, |u|\, \mathsf{else}\, |v| = |v| \\
|\mathsf{irr}_a| & : & |u| \overset{\mathsf{irr}_{|a|}}{=} |v| \\
|(x : A) \to b| & := & (x : |A|) \to |b| \\
|\lambda x.t| & := & \lambda x.|t| \\
|t \mathbin{@} x| & := & |t| \mathbin{@} x \\
|\pi\beta| & : & |(\lambda x.t) \mathbin{@} x| = |\lambda x.t| \mathbin{@} x = (\lambda x.|t|) \mathbin{@} x = |t| \\
|\pi\eta| & : & |\lambda x.t \mathbin{@} x| = \lambda x.|t \mathbin{@} x| = \lambda x.|t| \mathbin{@} x = |t| \\
|(u, v)| & := & (|u|, |v|) \\
|\mathsf{pr}_0\, t| & := & \mathsf{pr}_0\, |t| \\
|\mathsf{pr}_1\, t| & := & \mathsf{pr}_1\, |t| \\
|\sigma\beta_0| & : & |\mathsf{pr}_0\, (u, v)| = \mathsf{pr}_0\, |(u, v)| = \mathsf{pr}_0\, (|u|, |v|) = |u| \\
|\sigma\beta_1| & : & |\mathsf{pr}_1\, (u, v)| = \mathsf{pr}_1\, |(u, v)| = \mathsf{pr}_1\, (|u|, |v|) = |v| \\
|\sigma\eta| & : & |(\mathsf{pr}_0\, t, \mathsf{pr}_1\, t)| = (|\mathsf{pr}_0\, t|, |\mathsf{pr}_1\, t|) = (\mathsf{pr}_0\, |t|, \mathsf{pr}_1\, |t|) = |t| \\
|\top| & := & \top \\
|\mathsf{tt}| & := & \mathsf{tt} \\
|\bot| & := & \bot \\
|\mathsf{abort}\, t| & := & \mathsf{abort}\, |t|
\end{array}
$$

$$
\begin{array}{lll}
x^\sim & := & x_{01} \\
(t[\sigma])^\sim & := & t^\sim[\sigma^{01}] \\
(\lambda x.t)^\sim & := & \lambda x_0\, x_1\, x_{01}.t^\sim \\
(t \mathbin{@} x)^\sim & := & t^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} \\
\Pi\beta^\sim & : & ((\lambda x.t) \mathbin{@} x)^\sim = (\lambda x.t)^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = \\
 & & (\lambda x_0\, x_1\, x_{01}.t^\sim) \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = t^\sim \\
\Pi\eta^\sim & : & (\lambda x.t \mathbin{@} x)^\sim = \lambda x_0\, x_1\, x_{01}.(t \mathbin{@} x)^\sim = \\
 & & \lambda x_0\, x_1\, x_{01}.t^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = t^\sim \\
(u, v)^\sim & := & (u^\sim, v^\sim) \\
(\mathsf{pr}_0\, t)^\sim & := & \mathsf{pr}_0\, t^\sim \\
(\mathsf{pr}_1\, t)^\sim & := & \mathsf{pr}_1\, t^\sim \\
\Sigma\beta_0{}^\sim & : & (\mathsf{pr}_0\, (u, v))^\sim = \mathsf{pr}_0\, (u, v)^\sim = \mathsf{pr}_0\, (u^\sim, v^\sim) = u^\sim \\
\Sigma\beta_1{}^\sim & : & (\mathsf{pr}_1\, (u, v))^\sim = \mathsf{pr}_1\, (u, v)^\sim = \mathsf{pr}_1\, (u^\sim, v^\sim) = v^\sim
\end{array}
$$

$$
\begin{array}{lll}
\Sigma\eta^{\sim} & : & (\mathsf{pr}_0\, t, \mathsf{pr}_1\, t)^{\sim} = ((\mathsf{pr}_0\, t)^{\sim}, (\mathsf{pr}_1\, t)^{\sim}) = (\mathsf{pr}_0\, t^{\sim}, \mathsf{pr}_1\, t^{\sim}) = t^{\sim} \\
\mathsf{true}^{\sim} & := & \mathsf{tt} \\
\mathsf{false}^{\sim} & := & \mathsf{tt} \\
(\mathsf{if}\, t\, \mathsf{then}\, u\, \mathsf{else}\, v)^{\sim} & := & \\
\multicolumn{3}{l}{\quad \mathsf{if}\, t[0]\, \mathsf{then}\, (\mathsf{if}\, t[1]\, \mathsf{then}\, u^{\sim}\, \mathsf{else}\, \mathsf{abort}\, t^{\sim})\, \mathsf{else}\, (\mathsf{if}\, t[1]\, \mathsf{then}\, \mathsf{abort}\, t^{\sim}\, \mathsf{else}\, v^{\sim})} \\
\mathsf{Bool}\beta\mathsf{true}^{\sim} & : & (\mathsf{if}\, \mathsf{true}\, \mathsf{then}\, u\, \mathsf{else}\, v)^{\sim} = u^{\sim} \\
\mathsf{Bool}\beta\mathsf{false}^{\sim} & : & (\mathsf{if}\, \mathsf{false}\, \mathsf{then}\, u\, \mathsf{else}\, v)^{\sim} = v^{\sim} \\
\mathsf{irr}_a{}^{\sim} & : & u^{\sim} = \mathsf{tt} = v^{\sim} \\
((x : A) \to b)^{\sim} & = & \\
\multicolumn{3}{l}{\quad \big(\lambda f_0\, x_1.\mathsf{pr}_0\, (b^{\sim}[x_0 \mapsto \mathsf{coe}^*_A\, x_1, x_{01} \mapsto \mathsf{coh}^*_A\, x_1])\, @ (f_0\, @\, \mathsf{coe}^*_A\, x_1),} \\
\multicolumn{3}{l}{\quad \lambda f_1\, x_0.\mathsf{pr}_1\, (b^{\sim}[x_1 \mapsto \mathsf{coe}_A\, x_0, x_{01} \mapsto \mathsf{coh}_A\, x_0])\, @ (f_1\, @\, \mathsf{coe}_A\, x_0)\big)} \\
(\lambda x.t)^{\sim} & := & \mathsf{tt} \\
(t\, @\, x)^{\sim} & := & \mathsf{tt} \\
\pi\beta^{\sim} & : & ((\lambda x.t)\, @\, x)^{\sim} = \mathsf{tt} = t^{\sim} \\
\pi\eta^{\sim} & : & (\lambda x.t\, @\, x)^{\sim} = \mathsf{tt} = t^{\sim} \\
(u, v)^{\sim} & := & \mathsf{tt} \\
(\mathsf{pr}_0\, t)^{\sim} & := & \mathsf{tt} \\
(\mathsf{pr}_1\, t)^{\sim} & := & \mathsf{tt} \\
\sigma\beta_0{}^{\sim} & : & (\mathsf{pr}_0\, (u, v))^{\sim} = \mathsf{tt} = u^{\sim} \\
\sigma\beta_1{}^{\sim} & : & (\mathsf{pr}_1\, (u, v))^{\sim} = \mathsf{tt} = v^{\sim} \\
\sigma\eta^{\sim} & : & (\mathsf{pr}_0\, t, \mathsf{pr}_1\, t)^{\sim} = \mathsf{tt} = t^{\sim} \\
\top^{\sim} & := & (\lambda x.x, \lambda x.x) \\
\mathsf{tt}^{\sim} & := & \mathsf{tt} \\
\bot^{\sim} & := & (\lambda x.x, \lambda x.x) \\
(\mathsf{abort}\, t)^{\sim} & := & \mathsf{abort}\, (t[0_\Gamma])
\end{array}
$$

5 $\mathsf{MLTT}^{\mathsf{id+funext+propext}}_{\mathsf{Prop}}$

This type theory extends $\mathsf{MLTT}_{\mathsf{Prop}}$ (Section 2) with an identity type which goes into Prop and two axioms. The identity type has a definitional computation rule.

$$\frac{\Gamma \vdash_i A \qquad \Gamma \vdash_i t, t' : A}{\Gamma \vdash \mathsf{id}_A\, t\, t' : \mathsf{Prop}_i} \qquad \frac{\Gamma \vdash t : A}{\Gamma \vdash \mathsf{refl}_t : \underline{\mathsf{id}_A\, t\, t}}$$

$$\frac{\Gamma, x : A \vdash_i P \qquad \Gamma \vdash e : \underline{\mathsf{id}_A\, u\, v} \qquad \Gamma \vdash w : P[x \mapsto u]}{\Gamma \vdash \mathsf{transport}_{x.P}\, e\, w : P[x \mapsto v]}$$

$$\mathsf{id}\beta : \mathsf{transport}_{x.P}\, \mathsf{refl}_u\, w = w$$

Note that we have uniqueness of identity proofs (UIP), hence we can derive the eliminator J from $\mathsf{transport}$, there is no need to state it.

We have two extra rules: functional extensionality and propositional extensionality.

$$\frac{\Gamma \vdash t, t' : (x : A) \to B \qquad \Gamma, x : A \vdash e : \underline{\mathsf{id}_B\,(t \mathbin{@} x)\,(t' \mathbin{@} x)}}{\Gamma \vdash \mathsf{funext}\, e : \underline{\mathsf{id}_{(x:A)\to B}\, t\, t'}}$$

$$\frac{\Gamma \vdash a, a' : \mathsf{Prop} \qquad \Gamma \vdash w : \underline{(a \to a') \times (a' \to a)}}{\Gamma \vdash \mathsf{propext}\, w : \underline{\mathsf{id}_{\mathsf{Prop}}\, a\, a'}}$$

The setoid translation (Section 4) justifies these extra rules. This means that we can extend the domain of the setoid translation from $\mathsf{MLTT}_{\mathsf{Prop}}$ to $\mathsf{MLTT}_{\mathsf{Prop}}^{\mathsf{id}+\mathsf{funext}+\mathsf{propext}}$. The extra rules all introduce terms (and term equalities), so to extend the translation, we need to implement the two operations on terms for these extra rules. The $|-|$ operation is given as follows.

$$\begin{aligned}
|\mathsf{id}_A\, t\, t'| &:= A^{\sim}[\mathsf{R}_\Gamma]\,|t|\,|t'| \\
|\mathsf{refl}_t| &:= \mathsf{R}_A\,|t| \\
|\mathsf{transport}_{x.P}\, e\, w| &:= \mathsf{coe}_P[\mathsf{R}_\Gamma,\, x_0 \mapsto |u|,\, x_1 \mapsto |v|,\, x_{01} \mapsto |e|]\,|w| \\
|\mathsf{id}\beta| &: \quad |\mathsf{transport}_{x.P}\, \mathsf{refl}_u\, w| = \\
&\qquad \mathsf{coe}_P[\mathsf{R}_\Gamma,\, x_0 \mapsto |u|,\, x_1 \mapsto |u|,\, x_{01} \mapsto \mathsf{R}_A\,|u|]\,|w| = \\
&\qquad (\mathsf{coe}_P[\mathsf{R}_{\Gamma,x:A}]\,z)[x \mapsto |u|,\, z \mapsto |w|] \overset{\mathsf{coeR}_P}{=} |w| \\
|\mathsf{funext}\, e| &:= \lambda x_0\, x_1\, x_{01}.\mathsf{T}_B\,(\mathsf{R}_{\Gamma,x:A} \circ (x \mapsto x_0))\,\mathsf{R}_\Gamma \\
&\quad (|t| \mathbin{@} x_0)\,(|t'| \mathbin{@} x_0)\,(|t'| \mathbin{@} x_1)\,(|e|[x \mapsto x_0])\,(t'^{\sim}[\mathsf{R}_\Gamma] \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01}) \\
|\mathsf{propext}\, w| &:= |w|
\end{aligned}$$

The identity type of A is $A^{\sim}$ substituted by the R_Γ, reflexivity is R for the given type, $\mathsf{transport}$ comes from coercion and its computation rule from the computation rule of coercion. Propositional extensionality is definitional. We stated functional extensionality without using a heterogeneous equality, this is why it needs transitivity: first we show that $|t| \mathbin{@} x_0$ is equal to $|t'| \mathbin{@} x_0$ using $|e|$, then we use the fact that t' respects equality to show that this is equal to $|t'| \mathbin{@} x_1$.

The $-^{\sim}$ operation is implemented as follows.

$$\begin{aligned}
(\mathsf{id}_A\, t\, t')^{\sim} &:= \big(\lambda z.\mathsf{T}^3{}_A\,\mathsf{S}_\Gamma\,(\mathsf{R}_\Gamma \circ 0_\Gamma)\,\mathsf{id}_{\Gamma^{01}}\,(|t|[1_\Gamma])\,(|t|[0_\Gamma])\,(|t'|[0_\Gamma])\,(|t'|[1_\Gamma]) \\
&\qquad \big(\mathsf{S}_A\,(|t|[0_\Gamma])\,(|t|[1_\Gamma])\,t^{\sim}\big)\,z\,t'^{\sim} \\
&\quad \lambda z.\mathsf{T}^3{}_A\,\mathsf{id}_{\Gamma^{01}}\,(\mathsf{R}_\Gamma \circ 0_\Gamma)\,\mathsf{S}_\Gamma\,(|t|[0_\Gamma])\,(|t|[1_\Gamma])\,(|t'|[1_\Gamma])\,(|t'|[0_\Gamma]) \\
&\qquad t^{\sim}\,z\,\big(\mathsf{S}_A\,(|t'|[0_\Gamma])\,(|t'|[1_\Gamma])\,t'^{\sim}\big)\big) \\
\mathsf{refl}_t{}^{\sim} &:= \mathsf{tt} \\
(\mathsf{transport}_{x.P}\, e\, w)^{\sim} &:= \mathsf{let} \\
&\quad x_0 \mapsto \mathsf{coe}_P\big[\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_0 \mapsto |u|[0_\Gamma],\, x_1 \mapsto |v|[0_\Gamma],\, x_{01} \mapsto |e|[0_\Gamma]\big]\,(|w|[0_\Gamma]), \\
&\quad x_1 \mapsto |w|[0_\Gamma], \\
&\quad x_2 \mapsto |w|[1_\Gamma], \\
&\quad x_3 \mapsto \mathsf{coe}_P\big[\mathsf{R}_\Gamma \circ 1_\Gamma,\, x_0 \mapsto |u|[1_\Gamma],\, x_1 \mapsto |v|[1_\Gamma],\, x_{01} \mapsto |e|[1_\Gamma]\big]\,(|w|[1_\Gamma])\,\mathsf{in} \\
&\quad x_{01} \mapsto \mathsf{S}_P\big[\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_0 \mapsto |u|[0_\Gamma],\, x_1 \mapsto |v|[0_\Gamma],\, x_{01} \mapsto |e|[0_\Gamma]\big]\,x_1\,x_0
\end{aligned}$$

$$
\begin{array}{l}
\quad \big(\mathsf{coe}_P\big[\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_0 \mapsto |u|[0_\Gamma],\, x_1 \mapsto |v|[0_\Gamma],\, x_{01} \mapsto |e|[0_\Gamma]\big]\, x_1\big), \\
x_{12} \mapsto w^\sim, \\
x_{23} \mapsto \mathsf{coh}_P\big[\mathsf{R}_\Gamma \circ 1_\Gamma,\, x_0 \mapsto |u|[1_\Gamma],\, x_1 \mapsto |v|[1_\Gamma],\, x_{01} \mapsto |e|[1_\Gamma]\big]\, (|w|[1_\Gamma])\, \mathsf{in} \\
\mathsf{T}^3{}_P\, \big(\mathsf{S}_{\Gamma,x:A} \circ \big(\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_0 \mapsto |u|[0_\Gamma],\, x_1 \mapsto |v|[0_\Gamma],\, x_{01} \mapsto |e|[0_\Gamma]\big)\big)\, \mathsf{id}_{\Gamma^{01}} \\
\quad \big(\mathsf{R}_\Gamma \circ 1_\Gamma,\, x_0 \mapsto |u|[1_\Gamma],\, x_1 \mapsto |v|[1_\Gamma],\, x_{01} \mapsto |e|[1_\Gamma]\big)\, x_0\, x_1\, x_2\, x_3\, x_{01}\, x_{12}\, x_{23}
\end{array}
$$

$$
\begin{array}{lll}
\mathsf{id}\beta^\sim & : & (\mathsf{transport}_{x.P}\, \mathsf{refl}_u\, w)^\sim \overset{\mathsf{irr}_{(P[x\mapsto u])^\sim\, (|w|[0_\Gamma])\, (|w|[0_\Gamma])}}{=} w^\sim \\
(\mathsf{funext}\, e)^\sim & := & \mathsf{tt} \\
(\mathsf{propext}\, w)^\sim & := & \mathsf{tt}
\end{array}
$$

For identity, we need a logical equivalence between $|\mathsf{id}_A\, t\, t'|[0_\Gamma]$ and $|\mathsf{id}_A\, t\, t'|[1_\Gamma]$. The functions are given by using transitivity on $t^\sim$, the input identity and $t'^\sim$. For $\mathsf{transport}$, we need a heterogeneous identity between two coercions. We use transitivity on coherences (x_{01} and x_{23}) and congruence of the original term (x_{12}).

Note that if we had a universe of setoids (where identity is equality of codes) we would need to add an inductive-recursive universe of setoids to the syntax of $\mathsf{MLTT}_{\mathsf{Prop}}$ to justify it. Thus all the structure of the setoid translation would be included in the target of the translation.

6 SeTT: from translation to theory

A naive way to turn the setoid syntactic translation into a type theory would be adding the specification of the translation (Section 4.1) as derivation rules to $\mathsf{MLTT}_{\mathsf{Prop}}$ and adding the implementation (Section 4.2) as definitional equalities. However this type theory would not have an identity type: for each type A, its supposed identity type would be $A^\sim[\mathsf{R}_\Gamma]\, x_0\, x_1$ which is in a different context, $|\Gamma|$ instead of Γ.

The contexts $|\Gamma|$ and Γ are almost the same: the only difference is that $|\Gamma|$ has the extra components for functions saying that they respect the identity type. Our solution to this problem is *adding this extra component to the syntax*: in SeTT, in addition to $@$, functions have an extra eliminator ap ("apply path" in homotopy type theory, sometimes also called cong for congruence).

$$
\frac{\Gamma \vdash t : (x : A) \to B}{\Gamma, x_0\, x_1 : A, x_{01} : \underline{A^\sim[\mathsf{R}_\Gamma]\, x_0\, x_1} \vdash \mathsf{ap}\, t\, x_0\, x_1\, x_{01} : \underline{B^\sim[\mathsf{R}_\Gamma]\, (t\, @\, x_0)\, (t\, @\, x_1)}}
$$

We know how to compute congruence when the function is introduced by a λ:

$$
\Pi\beta_{\mathsf{ap}} : \mathsf{ap}\, (\lambda x.t)\, x_0\, x_1\, x_{01} = t^\sim[\mathsf{R}_\Gamma]
$$

Note that these rules refer to $^\sim$ and other components of the setoid translation, hence we are not able to simply add this rule to $\mathsf{MLTT}_{\mathsf{Prop}}$: we need to also add *syntax* for the setoid translation.

As described above, these are added as extra rules to $\mathsf{MLTT}_{\mathsf{Prop}}$ with the exception of $|-|$ which is not needed anymore. As a consequence, we need some adjustments to handle Π types.

The extra derivation rules of SeTT compared to $\mathsf{MLTT}_{\mathsf{Prop}}$ are the above two rules for ap and $\Pi\beta_{\mathsf{ap}}$ and the following.

$$\frac{\vdash \Gamma}{\begin{array}{l}\vdash \Gamma^{01}\\ k_\Gamma \;\; : \Gamma^{01} \Rightarrow \Gamma\\ \mathsf{R}_\Gamma \;\; : \Gamma \Rightarrow \Gamma^{01}\\ k\mathsf{R}_\Gamma : k_\Gamma \circ \mathsf{R}_\Gamma = \mathsf{id}_\Gamma\\ \mathsf{S}_\Gamma \;\; : \Gamma^{01} \Rightarrow \Gamma^{01}\\ k\mathsf{S}_\Gamma : k_\Gamma \circ \mathsf{S}_\Gamma = (1-k)_\Gamma\end{array}}$$

$$\frac{\begin{array}{c}\vdash \Gamma\\ \rho_{01}\,\rho_{12} : \Theta \Rightarrow \Gamma^{01}\\ 1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12}\end{array}}{\begin{array}{l}\mathsf{T}_\Gamma\,\rho_0\,\rho_1 : \Theta \Rightarrow \Gamma^{01}\\ 0\mathsf{T}_\Gamma : 0_\Gamma \circ \mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12} = 0_\Gamma \circ \rho_{01}\\ 1\mathsf{T}_\Gamma : 1_\Gamma \circ \mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12} = 1_\Gamma \circ \rho_{12}\end{array}}$$

$$\frac{\begin{array}{l}\vdash \Gamma\\ \rho\,\rho' : \Theta \Rightarrow \Gamma^{01}\\ \forall k.k_\Gamma \circ \rho = k_\Gamma \circ \rho'\end{array}}{\mathsf{irr}_\Gamma : \rho = \rho'}$$

$$\frac{\sigma : \Gamma \Rightarrow \Delta}{\begin{array}{ll}\sigma^{01} & : \Gamma^{01} \Rightarrow \Delta^{01}\\ \mathsf{nat}k_\sigma & : k_\Delta \circ \sigma^{01} = \sigma \circ k_\Gamma\end{array}}$$

$$\frac{\Gamma \vdash_i A}{\begin{array}{l}\Gamma^{01}, x_0 : A[0_\Gamma], x_1 : A[1_\Gamma] \vdash A^\sim\,x_0\,x_1 : \mathsf{Prop}_i\\ \Gamma, x : A \vdash \mathsf{R}_A\,x : \underline{A^\sim[\mathsf{R}_\Gamma]\,x\,x}\\ (\Gamma, x : A)^{01} \vdash \mathsf{S}_A\,x_0\,x_1\,x_{01} : \underline{A^\sim[\mathsf{S}_\Gamma]\,x_1\,x_0}\\ \Gamma^{01}, x_0 : A[0_\Gamma] \vdash \mathsf{coe}_A\,x_0 : A[1_\Gamma]\\ \Gamma^{01}, x_0 : A[0_\Gamma] \vdash \mathsf{coh}_A\,x_0 : \underline{A^\sim\,x_0\,(\mathsf{coe}_A\,x_0)}\\ \mathsf{coeR}_A : \mathsf{coe}_A[\mathsf{R}_\Gamma]\,x = x\end{array}}$$

$$\frac{\begin{array}{l}\Gamma \vdash_i A\\ \rho_{01}\,\rho_{12} : \Theta \Rightarrow \Gamma^{01}\\ 1_\Gamma \circ \rho_{01} = 0_\Gamma \circ \rho_{12}\\ \Theta \vdash t_0 : A[0_\Gamma \circ \rho_{01}]\\ \Theta \vdash t_1 : A[1_\Gamma \circ \rho_{01}]\\ \Theta \vdash t_2 : A[1_\Gamma \circ \rho_{12}]\\ \Theta \vdash t_{01} : \underline{A^\sim[\rho_{01}]\,t_0\,t_1}\\ \Theta \vdash t_{12} : \underline{A^\sim[\rho_{12}]\,t_1\,t_2}\end{array}}{\Theta \vdash \mathsf{T}_A\,\rho_{01}\,\rho_{12}\,t_0\,t_1\,t_2\,t_{01}\,t_{12} : \underline{A^\sim[\mathsf{T}_\Gamma\,\rho_{01}\,\rho_{12}]\,t_0\,t_2}}$$

$$\frac{\Gamma \vdash t : A}{\Gamma^{01} \vdash t^\sim : \underline{A^\sim\,(t[0_\Gamma])\,(t[1_\Gamma])}}$$

Note that these rules are the same as in the specification of the setoid translation (Section 4.1) except we removed all the $|-|$s. We also add the implementation of the translation (Section 4.2) as definitional equality rules. Again, we remove the $|-|$s and adjust the rules for Π. We only list those rules which have to be adjusted for Π and the extra rules for ap and $\Pi\beta_{\mathsf{ap}}$.

$$\begin{array}{ll}((x : A) \to B)^\sim\,f_0\,f_1 = (x_0 : A[0_\Gamma])(x_1 : A[1_\Gamma])(x_{01} : \underline{A^\sim\,x_0\,x_1}) \to & \\ \qquad B^\sim\,(f_0 \mathbin{@} x_0)\,(f_1 \mathbin{@} x_1) & \\ \mathsf{R}_{(x:A)\to B}\,z \qquad = \lambda x_0\,x_1\,x_{01}.\mathsf{ap}\,z\,x_0\,x_1\,x_{01} & \end{array}$$

$$
\begin{aligned}
&\mathsf{S}_{(x:A)\to B}\, f_0\, f_1\, f_{01} \quad = \\
&\quad \lambda x_0\, x_1\, x_{01}.\mathsf{S}_B[x_0 \mapsto x_1,\, x_1 \mapsto x_0,\, x_{01} \mapsto \mathsf{S}_A[\mathsf{S}_\Gamma]\, x_0\, x_1\, x_{01}] \\
&\quad (f_0 \mathbin{@} x_1)\, (f_1 \mathbin{@} x_0)\, (f_{01} \mathbin{@} x_1 \mathbin{@} x_0 \mathbin{@} \mathsf{S}_A[\mathsf{S}_\Gamma]\, x_0\, x_1\, x_{01}) \\
&\mathsf{coe}_{(x:A)\to B}\, f_0 \quad = \lambda x_1.\mathsf{coe}_B[x_0 \mapsto \mathsf{coe}_A^*\, x_1, x_{01} \mapsto \mathsf{coh}_A^*\, x_1]\, (f_0 \mathbin{@} \mathsf{coe}_A^*\, x_1) \\
&\mathsf{coh}_{(x:A)\to B}\, f_0 \quad = \lambda x_0\, x_1\, x_{01}.\mathsf{let}\, x_2 \mapsto \mathsf{coe}_A^*\, x_1,\, x_{12} \mapsto \mathsf{coh}_A[\mathsf{S}_\Gamma]\, x_1\, \mathsf{in} \\
&\quad \mathsf{let}\, x_{02} \mapsto \mathsf{T}_A\, \mathsf{id}_{\Gamma^{01}}\, \mathsf{S}_\Gamma\, x_0\, x_1\, x_2\, x_{01}\, x_{12},\, x_{21} \mapsto \mathsf{coh}_A^*\, x_1\, \mathsf{in} \\
&\quad \mathsf{T}_B\, (\mathsf{R}_\Gamma \circ 0_\Gamma,\, x_1 \mapsto x_2,\, x_{01} \mapsto x_{02})\, (x_0 \mapsto x_2, x_{01} \mapsto x_{21}) \\
&\qquad (f_0 \mathbin{@} x_0)\, (f_0 \mathbin{@} x_2)\, (\mathsf{coe}_B[x_0 \mapsto x_2,\, x_{01} \mapsto x_{21}]\, (f_0 \mathbin{@} x_2)) \\
&\qquad (\mathsf{ap}\, f_0\, x_0\, x_2\, x_{02})\, (\mathsf{coh}_B[x_0 \mapsto x_2, x_{01} \mapsto x_{21}]\, (f_0 \mathbin{@} x_2)) \\
&\mathsf{coeR}_{(x:A)\to B} \quad : \quad \mathsf{coe}_{(x:A)\to B}[\mathsf{R}_\Gamma]\, f_0 = \\
&\quad \lambda x_1.\mathsf{coe}_B\left[\mathsf{R}_\Gamma, x_0 \mapsto \mathsf{coe}_A^*[\mathsf{R}_\Gamma]\, x_1, x_{01} \mapsto \mathsf{coh}_A^*[\mathsf{R}_\Gamma]\, x_1\right]\, (f_0 \mathbin{@} \mathsf{coe}_A^*[\mathsf{R}_\Gamma]\, x_1) \overset{\mathsf{coeR}_A}{=} \\
&\quad \lambda x.\mathsf{coe}_B[\mathsf{R}_{\Gamma,x:A}]\, (f_0 \mathbin{@} x) \overset{\mathsf{coeR}_B}{=} \lambda x.(f_0 \mathbin{@} x) = f_0 \\
&\mathsf{T}_{(x:A)\to B}\, \rho_{01}\, \rho_{12}\, t_0\, t_1\, t_2\, t_{01}\, t_{12} = \\
&\quad \lambda x_0\, x_2\, x_{02}.\mathsf{let}\, x_1 \mapsto \mathsf{coe}_A[\rho_{01}]\, x_0,\, x_{01} \mapsto \mathsf{coh}_A[\rho_{01}]\, x_0\, \mathsf{in} \\
&\qquad \mathsf{let}\, x_{12} \mapsto \mathsf{T}_A\, (\mathsf{S}_\Gamma \circ \rho_{01})\, (\mathsf{T}_\Gamma\, \rho_{01}\, \rho_{12})\, x_1\, x_0\, x_2\, (\mathsf{S}_A[\rho_{01}]\, x_0\, x_1\, x_{01})\, x_{02}\, \mathsf{in} \\
&\qquad \mathsf{T}_B\, \rho_{01}\, (\rho_{12},\, x_0 \mapsto x_1,\, x_1 \mapsto x_2,\, x_{01} \mapsto x_{12})\, (t_0 \mathbin{@} x_0)\, (t_1 \mathbin{@} x_1)\, (t_2 \mathbin{@} x_2) \\
&\qquad\quad (t_{01} \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01})\, (t_{12} \mathbin{@} x_1 \mathbin{@} x_2 \mathbin{@} x_{12}) \\
&(\lambda x.t)^\sim \quad = \lambda x_0\, x_1\, x_{01}.t^\sim \\
&(t \mathbin{@} x)^\sim \quad = t^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} \\
&\Pi\beta^\sim \quad : \quad ((\lambda x.t) \mathbin{@} x)^\sim = (\lambda x.t)^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = \\
&\qquad\quad (\lambda x_0\, x_1\, x_{01}.t^\sim) \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = t^\sim \\
&\Pi\eta^\sim \quad : \quad (\lambda x.t \mathbin{@} x)^\sim = \lambda x_0\, x_1\, x_{01}.(t \mathbin{@} x)^\sim = \\
&\qquad\quad \lambda x_0\, x_1\, x_{01}.t^\sim \mathbin{@} x_0 \mathbin{@} x_1 \mathbin{@} x_{01} = t^\sim \\
&(\mathsf{ap}\, t\, x_0\, x_1\, x_{01})^\sim \quad = \mathsf{tt} \\
&\Pi\beta_{\mathsf{ap}}{}^\sim \quad : \quad (\mathsf{ap}\, (\lambda x.t)\, x_0\, x_1\, x_{01})^\sim = \mathsf{tt} = (t^\sim[\mathsf{R}_\Gamma])^\sim
\end{aligned}
$$

We conjecture that we can justify the rules of setoid type theory by a syntactic translation into $\mathsf{MLTT}_{\mathsf{Prop}}$: we interpret the non-$\mathsf{MLTT}_{\mathsf{Prop}}$ parts of the syntax as if they were operations and not syntax and the $\mathsf{MLTT}_{\mathsf{Prop}}$-part by $|-|$.

7 Conclusions and further work

In this paper we showed an example of turning a model of type theory into a type theory. First we defined a syntactic translation corresponding to the setoid model which mapped contexts to contexts (with possibly extra structure). Then we added the translation itself as syntax to the theory. We had to make an adjustment to the theory to make the context of the identity type the same as the original context: this amounted to adding an additional congruence eliminator to the function space. We expect that similar adjustments need to be made for coinductive types. The justification of SeTT is left as further work, just as its extension with a universe of setoids and general inductive and coinductive types.

Extensional type theory (ETT) can be translated to $\mathsf{MLTT}^{\mathsf{id+funext+propext}}_{\mathsf{Prop}}$ [12, 18]. ETT is considered as an internal language for topoi [15]. SeTT could provide a more convenient language with decidability of definitional equality and thus a convenient type checking algorithm. In the future, we would like to investitage the exact relationship between setoid type theory and topoi. It seems plausible that the current SeTT can serve as an internal language for quasitopoi instead of topoi. We would need an open universe of propositions for a subobject classifier: when it is proven that all inhabitants of a type are identical, the type can be regarded as a proposition.

As further work, we mention the need for a generic method to turn (strict) models of type theory into syntactic translations.

References

[1] Thorsten Altenkirch. Extensional equality in intensional type theory. In *14th Symposium on Logic in Computer Science*, pages 412 – 420, 1999.

[2] Thorsten Altenkirch and Ambrus Kaposi. Type theory in type theory using quotient inductive types. In Rastislav Bodik and Rupak Majumdar, editors, *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, pages 18–29. ACM, 2016.

[3] Thorsten Altenkirch and Ambrus Kaposi. Towards a Cubical Type Theory without an Interval. In Tarmo Uustalu, editor, *21st International Conference on Types for Proofs and Programs (TYPES 2015)*, volume 69 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:27, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[4] Thorsten Altenkirch, Conor Mcbride, and Wouter Swierstra. Observational equality, now! In *PLPV '07: Proceedings of the 2007 workshop on Programming languages meets program verification*, pages 57–58. ACM, 2007.

[5] Jean-Philippe Bernardy, Patrik Jansson, and Ross Paterson. Proofs for free — parametricity for dependent types. *Journal of Functional Programming*, 22(02):107–152, 2012.

[6] Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. In *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, volume 26, pages 107–128, 2014.

[7] Simon Boulier, Pierre-Marie Pédrot, and Nicolas Tabareau. The next 700 syntactical models of type theory. In *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs*, CPP 2017, pages 182–194, New York, NY, USA, 2017. ACM.

[8] John Cartmell. Generalised algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209–243, 1986.

[9] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom. In Tarmo Uustalu, editor, *21st International Conference on Types for Proofs and Programs (TYPES 2015)*, volume 69 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:34, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[10] Peter Dybjer. Internal type theory. In *Lecture Notes in Computer Science*, pages 120–134. Springer, 1996.

[11] Hugo Herbelin. Syntactic investigations into cubical type theory. In José Espĺrito Santo and Luĺs Pinto, editors, *24th International Conference on Types for Proofs and Programs, TYPES 2018*. University of Minho, 2018.

[12] Martin Hofmann. Conservativity of equality reflection over intensional type theory. In *TYPES 95*, pages 153–164, 1995.

[13] Martin Hofmann. *Extensional concepts in intensional type theory*. Thesis. University of Edinburgh, Department of Computer Science, 1995.

[14] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. In *In Venice Festschrift*, pages 83–111. Oxford University Press, 1996.

[15] Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. Internal Universes in Models of Homotopy Type Theory. In Hélène Kirchner, editor, *3rd International Conference on Formal Structures for Computation and Deduction (FSCD 2018)*, volume 108 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:17, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[16] Nicolas Tabareau Matthieu Sozeau. Univalence for free, 2013. `http://hal.inria.fr/hal-00786589/en`.

[17] Conor McBride. Elimination with a motive. In *Selected Papers from the International Workshop on Types for Proofs and Programs*, TYPES '00, pages 197–216, Berlin, Heidelberg, 2002. Springer-Verlag.

[18] Nicolas Oury. *Extensionality in the calculus of constructions*, pages 278–293. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[19] The Univalent Foundations Program. Homotopy type theory: Univalent foundations of mathematics. Technical report, Institute for Advanced Study, 2013.

[20] Thomas Streicher. Investigations into intensional type theory. habilitation thesis, 1993.

[21] Nicolas Tabareau, Éric Tanter, and Matthieu Sozeau. Equivalences for Free. In *ICFP 2018 - International Conference on Functional Programming*, Saint-Louis, United States, September 2018.