An extensional Kleene realizability semantics for the Minimalist Foundation

Maria Emilia Maietti and Samuele Maschio[1]

1 Dipartimento di Matematica, University of Padova, Via Trieste, 63 - I-35121 Padova, Italy, {maietti,maschio}@math.unipd.it

Abstract

We build a Kleene realizability semantics for the two-level Minimalist Foundation **MF**, ideated by Maietti and Sambin in 2005 and completed by Maietti in 2009. Thanks to this semantics we prove that both levels of **MF** are consistent with the formal Church Thesis **CT**.

Since **MF** consists of two levels, an intensional one, called **mTT**, and an extensional one, called **emTT**, linked by an interpretation, it is enough to build a realizability semantics for the intensional level **mTT** to get one for the extensional one **emTT**, too. Moreover, both levels consists of type theories based on versions of Martin-Löf's type theory.

Our realizability semantics for **mTT** is a modification of the realizability semantics by Beeson in 1985 for extensional first order Martin-Löf's type theory with one universe. So it is formalized in Feferman's classical arithmetic theory of inductive definitions, called $\widehat{ID_1}$. It is called *extensional* Kleene realizability semantics since it validates extensional equality of type-theoretic functions **extFun**, as in Beeson's one.

The main modification we perform on Beeson's semantics is to interpret propositions, which are defined primitively in **MF**, in a proof-irrelevant way. As a consequence, we gain the validity of **CT**. Recalling that **extFun**+ **CT**+ **AC** are inconsistent over arithmetics with finite types, we conclude that our semantics does not validate the Axiom of Choice **AC** on generic types. On the contrary, Beeson's semantics does validate **AC**, being this a theorem of Martin-Löf's theory, but it does not validate **CT**. The semantics we present here seems to be the best approximation of Kleene realizability for the extensional level **emTT**. Indeed Beeson's semantics is not an option for **emTT** since **AC** on generic sets added to it entails the excluded middle.

1998 ACM Subject Classification F. 4. 1 Computability theory, Lambda calculus and related systems

Keywords and phrases Realizability, Type Theory, formal Church Thesis

Digital Object Identifier 10.4230/LIPIcs.TYPES.2014.XXX

1 Introduction

A foundation for mathematics should be called constructive only if the mathematics arising from it could be considered genuinely computable. One way to show this is to produce a realizability model of the foundation where arbitrary sets are interpreted as data types and functions between them are interpreted as programs. A key example is Kleene's realizability model for first-order Intuitionist Arithmetics validating the formal Church Thesis.

Here we will show how to build a realizability model for the *Minimalist Foundation*, for short **MF**, ideated by Maietti and Sambin in [13] and then completed by Maietti in [9], where it is explicit how to extract programs from its proofs. In particular we show that **MF** is consistent with the Church Thesis, for short **CT**. This result is part of a project to know to what extent **MF** enjoys the same properties as Heyting arithmetics.

The Minimalist Foundation is intended to constitute a common core among the most relevant constructive and classical foundations. One of its novelties is that it consists of two levels: an intensional level, called **mTT**, which should make evident the constructive contents of mathematical proofs in terms of programs, and an extensional level, called **emTT**, formulated in a language close as much as possible to that of ordinary mathematics. Both intensional and extensional levels of **MF** consist of type systems based on versions of Martin-Löf's type theory with the addition of a primitive notion of propositions: the intensional one is based on [17] and the extensional one on [16]. Actually **mTT** can be considered a *predicative* version of Coquand's Calculus of Constructions [4].

To build a realizability model for the two-level Minimalist Foundation, it is enough to build it for its intensional level **mTT**. Indeed an interpretation for the extensional level **emTT** can be then obtained from an interpretation of **mTT** by composing this with the interpretation of **emTT** in a suitable setoid model of **mTT** as in [9] and analyzed in [11]. Moreover, since the interpretation of **CT** from the extensional level to the intensional one is equivalent to **CT** itself according to [9], a model showing consistency of **mTT** with **CT** can be turned into a model showing consistency of **emTT** with **CT**.

Here, we build a realizability model for **mTT**+ **CT** by suitably modifying Beeson's realizability semantics [2] for the extensional version of first order Martin-Löf's type theory with one universe [16]. So, as Beeson's semantics our model is based on Kleene realizability semantics of intuitionistic arithmetics and it is formalized in Feferman's classical arithmetic theory of inductive definitions, called $\widehat{ID_1}$ ([5]). The theory $\widehat{ID_1}$ is formulated in the language of second-order arithmetics and it consists of PA (Peano Arithmetic) plus the existence of some (not necessary the least) fix point for positive parameter-free arithmetical operators.

We call our Kleene realizability semantics *extensional* since it validates extensional equality of type-theoretic functions **extFun**, as Beeson's one.

The main modification we perform to Beeson's semantics is to interpret propositions, which are defined primitively in **MF**, in a proof-irrelevant way. More in detail we interpret **mTT**-sets as Beeson interpreted Martin-Löf's sets, propositions are interpreted as trivial quotients of Kleene realizability interpretation of intuitionistic connectives, and the universe of **mTT**-small propositions is interpreted as a suitable quotient of some fix point including all the codes of small propositions by using the technique Beeson adopted to interpret Martin-Löf's universe.

As a consequence in our model we gain the validity of **CT** but we loose the validity of the full Axiom of Choice **AC**. Instead in Beeson's semantics, **AC** is valid, being this a theorem of Martin-Löf's theory, but **CT** is not. All these results follow from the well known fact that **extFun**+ **CT**+ **AC** over arithmetics with finite types are inconsistent. Therefore in the presence of **extFun** as in our **emTT**, either one validates **CT** as we do here, or **AC** as in Beeson's semantics. Recalling that the addition of **AC** on generic sets in **emTT** entails the excluded middle, Beeson's semantics is not an option for **emTT**. Therefore the semantics we present here appears to be the best approximation of Kleene realizability for the extensional level **emTT**.

Actually a consistency proof for **emTT** with **CT** could also be obtained by interpreting this theory in the internal theory of Hyland's effective topos [7]. But here we have obtained a proof in a *predicative theory*, whilst classical, as $\widehat{ID_1}$. As a future work we intend to generalize the notion of tripos-to-topos construction in [8] in order to extract the categorical structure behind our realizability interpretation to the final goal of building a predicative effective topos.

2 The Minimalist Foundation

In [9] a two-level formal system, called **Minimalist Foundation**, for short **MF**, is completed following the design advocated in [13]. The two levels of **MF** are both given by a type theory à la Martin-Löf: the intensional level, called **mTT**, is an intensional type theory including aspects of Martin-Löf's one in [17] (and extending the set-theoretic version in [13] with collections), and its extensional level, called **emTT**, is an extensional type theory including aspects of extensional Martin-Löf's one in [16]. Then a quotient model of setoids à la Bishop [3, 6, 1, 19] over the intensional level is used in [9] to interpret the extensional level in the intensional one. A categorical study of this quotient model has been carried on in [11, 10, 12] and related to the construction of Hyland's effective topos [7, 8].

MF was ideated in [13] to be constructive and minimalist, that is compatible with (or interpretable in) most relevant constructive and classical foundations for mathematics in the literature. According to these desiderata, **MF** has the following peculiar features (for a more extensive description see also [14]):

- **MF has two types of entities: sets and collections.** This is a consequence of the fact that a minimalist foundation compatible with most of constructive theories in the literature, among which, for example, Martin-Löf's one in [17], should be certainly predicative and based on intuitionistic predicate logic, including at least the axioms of Heyting arithmetic. For instance it could be a many-sorted logic, such as Heyting arithmetic of finite types [20], where sorts, that we call *types*, include the basic sets we need to represent our mathematical entities. But in order to represent topology in an intuitionistic and predicative way, then **MF** needs to be equipped with two kinds of entities: sets and collections. Indeed, the *power of a non-empty set*, namely the discrete topology over a non-empty set, fails to be a set in a predicative foundation, and it is only a *collection*.
- **MF has two types of propositions.** This is a consequence of the previous characteristic. Indeed the presence of sets and collections, where the latter include the representation of power-collections of subsets, yields to distinguish two types of propositions to remain predicative: those closed under quantifications on sets, called *small propositions* in [9], from those closed under any kind of quantification, called *propositions* in [9]. This distinction is crucial in the definition of "subset of a set" we adopt in **MF**: a subset of a set A is indeed an equivalence class of small propositional functions from A.
- **MF has two types of functions.** As in Coquand's Calculus of Constructions [4], or Feferman's predicative theories [5], in **MF** we distinguish the notion of functional relation from that of type-theoretic function. In particular *in* **MF** *only type-theoretic functions between two sets form a set, while functional relations between two sets form generally a collection.*

  This restriction is crucial to make **MF** compatible with classical predicative theories as Feferman's predicative theories [5]. Indeed it is well-known that the *addition of the principle of excluded middle* can turn a predicative theory where functional relations between sets form a set, as Aczel's CZF or Martin-Löf's type theory, into an impredicative one where *power-collections become sets.*

2.1 The intensional level of the Minimalist Foundation

Here we describe the intensional level of the Minimalist Foundation in [9], which is represented by a dependent type theory called **mTT**. This type theory is written in the style of Martin-

Löf's type theory [17] by means of the following four kinds of judgements:

$$A\ type\ [\Gamma] \qquad A = B\ type\ [\Gamma] \qquad a \in A\ [\Gamma] \qquad a = b \in A\ [\Gamma]$$

that is the type judgement (expressing that something is a specific type), the type equality judgement (expressing when two types are equal), the term judgement (expressing that something is a term of a certain type) and the term equality judgement (expressing the *definitional equality* between terms of the same type), respectively, all under a context Γ.

The word *type* is used as a meta-variable to indicate four kinds of entities: collections, sets, propositions and small propositions, namely

$$type \in \{col, set, prop, prop_s\}$$

Therefore, in **mTT** types are actually formed by using the following judgements:

$$A\ set\ [\Gamma] \qquad D\ col\ [\Gamma] \qquad \phi\ prop\ [\Gamma] \qquad \psi\ prop_s\ [\Gamma]$$

saying that A is a set, that D is a collection, that ϕ is a proposition and that ψ is a small proposition.

Here, contrary to [9] where capital latin letters are used as meta-variables for all types, we use greek letters ψ, ϕ as meta-variables for propositions, we mostly use capital latin letters A, B as meta-variables for sets and capital latin letters C, D as meta-variables for collections.

As in the intensional version of Martin-Löf's type theory, in **mTT** there are two kinds of equality concerning terms: one is the definitional equality of terms of the same type given by the judgement

$$a = b \in A\ [\Gamma]$$

which is decidable, and the other is the propositional equality written

$$\mathsf{Id}(A, a, b)\ prop\ [\Gamma]$$

which is not necessarily decidable.

We now proceed by briefly describing the various kinds of types in **mTT**, starting from small propositions and propositions and then passing to sets and finally collections.

Small propositions in **mTT** include all the logical constructors of intuitionistic predicate logic with equality and quantifications restricted to sets:

$$\phi\ prop_s \ \equiv\ \ \bot \ |\ \ \phi \wedge \psi\ \ |\ \phi \vee \psi\ \ |\ \ \phi \to \psi\ \ |\ (\forall x \in A)\ \phi(x)\ \ |\ (\exists x \in A)\ \phi(x)\ \ |\ \ \mathsf{Id}(A, a, b)$$

provided that A is a set.

Then, *propositions* in **mTT** include all the logical constructors of intuitionistic predicate logic with equality and quantifications on all kinds of types, i.e. sets and collections. Of course, small propositions are also propositions.

$$\phi\ prop \ \equiv\ \ \phi\ prop_s\ |\ \phi \wedge \psi\ \ |\ \phi \vee \psi\ \ |\ \ \phi \to \psi\ \ |\ (\forall x \in D)\ \phi(x)\ \ |\ (\exists x \in D)\ \phi(x)\ \ |\ \ \mathsf{Id}(D, d, b)$$

In order to close sets under comprehension, for example to include the set of positive natural numbers $\{x \in \mathrm{N}\ \mid\ x \geq 1\}$, and to define operations on such sets, we need to think of propositions as types of their proofs: small propositions are seen as sets of their proofs while generic propositions are seen as collections of their proofs. That is, we add to **mTT** the following rules

$$\textbf{prop}_s\textbf{-into-set)}\ \ \frac{\phi\ prop_s}{\phi\ set} \qquad\qquad \textbf{prop-into-col)}\ \ \frac{\phi\ prop}{\phi\ col}$$

Before explaining the difference between the notion of set and collection we describe their constructors in **mTT**.

Sets in **mTT** are characterized as inductively generated types and they include the following:

$$A\ set \ \equiv\ \ \phi\ prop_s \mid N_0 \mid N_1 \mid N \mid List(A) \mid (\Sigma x \in A)\, B(x) \mid A + B \mid (\Pi x \in A)\, B(x)$$

where the notation N_0 stands for the empty set, N_1 for the singleton set, N for the set of natural numbers, $List(A)$ for the set of Lists on the set A, $(\Sigma x \in A)B(x)$ for the indexed sum of the family of sets $B(x)\ set\ [x \in A]$ indexed on the set A, $A + B$ for the disjoint sum of the set A with the set B, $(\Pi x \in A)B(x)$ for the product type of the family of sets $B(x)\ set\ [x \in A]$ indexed on the set A.

It is worth noting that the set N of the natural numbers is not present in a primitive way in **mTT** in [9] since its rules can be derived by putting $N \equiv List(N_1)$. Here we add it to the syntax of **mTT** because it plays a prominent role in realizability and we want to interpret it directly in $\widehat{ID_1}$ to avoid complications due to list encodings.

Finally, *collections* in **mTT** include the following types:

$$D\ col \ \equiv\ \ A\ set \ \mid\ \phi\ prop \ \mid\ \mathsf{props} \ \mid\ A \to \mathsf{props} \ \mid\ (\Sigma x \in D)\, E(x)$$

where props stands for the collection of (codes for) small propositions and $A \to \mathsf{props}$ for the collection of propositional functions of the set A, while $(\Sigma x \in D)\, E(x)$ stands for the indexed sum of the family of collections $E(x)\ col\ [x \in D]$ indexed on the collection D.
All sets are collections thanks to the following rule:

$$\textbf{set-into-col)}\ \ \frac{A\ set}{A\ col}$$

Note that the collection of small propositions props is defined here with codes à la Tarski as in [17], contrary to the version in [9], to make the interpretation easier to understand. Its rules are the following.
Elements of the collection of small propositions are generated as follows:

$$\text{Pr}_1)\ \ \widehat{\bot} \in \mathsf{props} \qquad\qquad \text{Pr}_2)\ \ \frac{p \in \mathsf{props}\quad q \in \mathsf{props}}{p \widehat{\vee} q \in \mathsf{props}}$$

$$\text{Pr}_3)\ \ \frac{p \in \mathsf{props}\qquad q \in \mathsf{props}}{p \widehat{\to} q \in \mathsf{props}} \qquad \text{Pr}_4)\ \ \frac{p \in \mathsf{props}\qquad q \in \mathsf{props}}{p \widehat{\wedge} q \in \mathsf{props}}$$

$$\text{Pr}_5)\ \ \frac{A\ set\quad a \in A\quad b \in A}{\widehat{\mathsf{Id}}(A, a, b) \in \mathsf{props}} \qquad \text{Pr}_6)\ \ \frac{p(x)\ \mathsf{props}\ \ [x \in B]\qquad B\ set}{\widehat{(\exists x \in B)}p(x) \in \mathsf{props}}$$

$$\text{Pr}_7)\ \ \frac{p(x) \in \mathsf{props}\ [x \in B]\qquad B\ set}{\widehat{(\forall x \in B)}p(x) \in \mathsf{props}}$$

Elements of the collection of small propositions can be decoded as small propositions via an operator as follows

$$\tau\text{-Pr)}\ \ \frac{p \in \mathsf{props}}{\tau(p)\ prop_s}$$

and this operator satisfies the following definitional equalities:

eq-Pr$_1$) $\tau(\widehat{\bot}) = \bot \, props_s$

eq-Pr$_2$) $\dfrac{p \in \mathsf{props_s} \quad q \in \mathsf{props_s}}{\tau(p \widehat{\vee} q) = \tau(p) \vee \tau(q) \, props_s}$

eq-Pr$_3$) $\dfrac{p \in \mathsf{props_s} \qquad q \in \mathsf{props_s}}{\tau(p \widehat{\to} q) = \tau(p) \to \tau(q) \, props_s}$

eq-Pr$_4$) $\dfrac{p \in \mathsf{props_s} \qquad q \in \mathsf{props_s}}{\tau(p \widehat{\wedge} q) = \tau(p) \wedge \tau(q) \, props_s}$

eq-Pr$_5$) $\dfrac{A \; set \quad a \in A \quad b \in A}{\tau(\, \widehat{\mathsf{Id}}(A, a, b)\,) = \mathsf{Id}(A, a, b) \, props_s}$

eq-Pr$_6$) $\dfrac{p(x) \; \mathsf{props_s} \; [x \in B] \qquad B \; set}{\tau(\widehat{(\exists x \in B)} p(x)) = (\exists x \in B) \tau(p(x)) \, props_s}$

eq-Pr$_7$) $\dfrac{p(x) \in \mathsf{props_s} \; [x \in B] \qquad B \; set}{\tau(\widehat{(\forall x \in B)} p(x)) = (\forall x \in B) \tau(p(x)) \, props_s}$

In the realizability interpretation of **mTT** we need to define a subset of natural numbers including codes of **mTT**-sets in order to define the subset of codes of small propositions closed under quantification on sets. The existence of such a subset of set codes says that the realizability interpretation is actually interpreting an extension of **mTT** with a collection of sets. In order to simplify the definition of the realizability interpretation, we interpret an extension of **mTT**, which we call **mTT**^{s}, with the addition of the collection Set of set codes whose related rules are the following. We don't give any elimination and conversion rule as those of universes à la Tarski in [17] since it would not be validated in the model (because we do not have least fix-points in $\widehat{ID_1}$).

Collection of sets

F-Se) $\mathsf{Set} \; col$

Elements of the collection of sets are generated as follows:

Se$_e$) $\widehat{N_0} \in \mathsf{Set}$ Se$_s$) $\widehat{N_1} \in \mathsf{Set}$ Se$_n$) $\widehat{N} \in \mathsf{Set}$

Se$_l$) $\dfrac{a \in \mathsf{Set}}{\widehat{List}(a) \in \mathsf{Set}}$ Se$_u$) $\dfrac{a \in \mathsf{Set} \quad b \in \mathsf{Set}}{a \widehat{+} b \in \mathsf{Set}}$

Se$_\Sigma$) $\dfrac{a(x) \; \mathsf{Set} \; [x \in B] \qquad B \; set}{\widehat{(\Sigma x \in B)} a(x) \in \mathsf{Set}}$ Se$_\Pi$) $\dfrac{a(x) \; \mathsf{Set} \; [x \in B] \qquad B \; set}{\widehat{(\Pi x \in B)} a(x) \in \mathsf{Set}}$

sp-i-s) $\dfrac{p \in \mathsf{props_s}}{p \in \mathsf{Set}}$

mTT can be viewed as a *predicative version* of the Calculus of Constructions [4], for short CoC. The main difference with respect to CoC is that **mTT** distinguishes between sets and collections in a way similar to the distinction between sets and classes in axiomatic set theory. However, all types of **mTT**, i.e. small propositions, propositions, sets and collections, are predicative entities in the sense that their elements can be generated in an inductive way by a finite number of rules. According to the notion of set in Bishop [3] and Martin-Löf [15], all **mTT**-types are actually sets, and in fact **mTT**-types can be interpreted as sets in the intensional version of Martin-Löf's type theory in [17]. The **mTT**-distinction between sets and collections, and the corresponding distinction between small propositions and propositions, is motivated by the need of distinguishing between predicative entities whose notion of element is a closed concept, and these are called sets, and those entities whose notion of element is an open concept, and these are called collections. The motivating idea is that a set is inductively generated by a finite number of rules whose associated inductive principle does not vary when the theory **mTT** is extended with new entities (sets, collections or propositions). On the contrary a collection is inductively generated by a finite number of rules which may vary when the theory is extended with new entities. Typical examples of collections are universes (of sets or propositions): if we extend the theory **mTT** with a new small proposition, then we need to add a new rule inserting this new small proposition in the collection of small propositions.

We recall from [13] that the distinction between propositions and sets is crucial to avoid the validity of choice principles.

Finally, it is worth noting that in **mTT** we restrict substitution term equality rules to explicit substitution term equality rules of the form

$$\text{sub)}\ \frac{\begin{array}{c} c(x_1,\dots,x_n)\in C(x_1,\dots,x_n)\ \ [x_1\in A_1,\ \dots,\ x_n\in A_n(x_1,\dots,x_{n-1})\,] \\ a_1=b_1\in A_1\ \ \dots\ \ a_n=b_n\in A_n(a_1,\dots,a_{n-1})\end{array}}{c(a_1,\dots,a_n)=c(b_1,\dots,b_n)\in C(a_1,\dots,a_n)}$$

in place of usual term equality rules preserving term constructions typical of Martin-Löf's type theory in [17]. This restriction, and in particular the absence of the so called ξ-rule of lambda-terms

$$\xi\ \frac{c=c'\in C\ [x\in B]}{\lambda x^B.c=\lambda x^B.c'\in(\Pi x\in B)C}$$

seems to be crucial to prove consistency of **mTT** with **AC+CT**, as advocated in [13], by means of a realizability semantics à la Kleene, but this is still an open problem (the realizability semantics given here does not help to solve this since it can not validate **AC** on all types). It is worth to recall from [9] that our restriction of term equality does not affect the possibility of adopting **mTT** as the intensional level of a two-level constructive foundation as intended in [13]. Indeed the term equality rules of **mTT** suffice to interpret an extensional level including extensional equality of functions, as that represented by **emTT**, by means of the quotient model described in [9] and studied abstractly in [11, 10, 12].

2.2 The extensional level of the Minimalist Foundation

Here we briefly describe the extensional level **emTT** of the Minimalist Foundation. This is an extensional dependent type theory extending extensional Martin-Löf's type theory in [16] with primitive (proof-irrelevant) propositions, power-collections and quotients.

The rules of **emTT** are formulated by using the same kinds of judgements used for **mTT**. The main peculiar characteristics of **emTT** in comparison to **mTT** are the following.

1. A primary difference between **emTT** and **mTT** is the usual difference between the so called intensional version of Martin-Löf's type theory [17] and its extensional one in [16] and this is the fact that the definitional equality of terms

   $$a=b\in A\ [\Gamma]$$

   is no longer *decidable* in **emTT** as it is in the intensional **mTT**. This is in turn due to the fact that the propositional equality of **emTT** as that of [16], called $\mathsf{Eq}(A,a,b)$, is extensional in the sense that the provability of $\mathsf{Eq}(A,a,b)\ [\Gamma]$ in **emTT** is equivalent to the derivation of the judgement $a=b\in A\ [\Gamma]$. Instead, in **mTT** only the derivation of the definitional equality judgement $a=b\in A\ [\Gamma]$ implies internally the provability of the intensional propositional equality $\mathsf{Id}(A,a,b)\ [\Gamma]$ under a generic context.
2. Another peculiar feature of **emTT** employs the distinction between propositions and sets: this is the addition of proof-irrelevance for propositions captured by the following rules

$$\textbf{prop-mono)}\ \frac{\phi\ prop\ [\Gamma]\qquad p\in\phi\ [\Gamma]\quad q\in\phi\ [\Gamma]}{p=q\in\phi\quad[\Gamma]}\qquad\qquad \textbf{prop-true)}\ \frac{\phi\ prop\qquad\quad p\in\phi}{\mathsf{true}\in\phi}$$

saying that a proof of a proposition is *unique* and equal to a canonical proof term called true. Of course, these rules can not be added to an extensional theory identifying propositions with sets as Martin-Löf's one in [16], because they would trivialize all constructors. Moreover, these rules are not present in the intensional level **mTT** because proof-irrelevance is a typical extensional condition. Indeed, **emTT**-propositions can be thought of as quotients of intensional propositions under the trivial equivalence relation between proofs.

3. Other key differences between the type theories **mTT** and **emTT** are the addition in **emTT** of quotient sets
$$A/\rho \ set\ [\Gamma]$$
provided that ρ is a small equivalence relation $\rho\ prop_s\ [x \in A, y \in A]$ on the set A, and the addition of the power-collection of the singleton and of the power-collection of a generic set A
$$\mathcal{P}(1) \qquad\qquad A \to \mathcal{P}(1)$$
4. A further difference between the type theories **mTT** and **emTT** concerns the equality rules between terms. Indeed in **emTT** equality rules between terms are the usual ones typical of an extensional type theory in [16] preserving all term constructors. In particular, equality of lambda-functions is extensional, namely it is possible to prove
$$(\forall x \in A)\mathsf{Eq}(\,B(x), f(x)\,,\, g(x)) \;\to\; \mathsf{Eq}(\,(\Pi x \in A)B(x)\,,\ \lambda x.f(x)\ ,\ \lambda x.g(x)\,)$$
This proposition is not necessarily provable at the intensional level **mTT** when substituting the extensional propositional equality $\mathsf{Eq}(A, a, b)$ with the intensional one $\mathsf{Id}(A, a, b)$.

We end by recalling from [9] that *a model for* **mTT** *can be turned into a model for* **emTT** *by using the interpretation of* **emTT** *into* **mTT** *described in* [9]. Therefore in the following we are going to define a realizability interpretation just for **mTT**, to get one also for **emTT**.

2.3 Untyped syntax of $\mathbf{mTT}^s$

Usually in type theory the syntax is introduced *in fieri*; for example terms are introduced typically after deriving some conditions or constraints which are required to define them. However for semantical purposes it looks more convenient to present the syntax *a priori in a partial way* by eliminating parts of usual restrictions.

Therefore, since we want to define a realizability interpretation for $\mathbf{mTT}^s$, we introduce here the syntax of all $\mathbf{mTT}^s$-type and term constructors in a partial way and we refer the reader to look at [9] for all the **mTT**-rules. Then we will define a partial interpretation for terms of our *extended* syntax and check that this interpretation is well defined in case the constraints for introducing them are validated by the model.

▶ **Definition 1.** Let $[\underline{x}]$ be a context, i. e. $[\underline{x}] = [x_1, ..., x_\mathsf{n}]$ is a possibly empty list of distinct variables. *Terms, small propositions, sets, propositions and collections* in context are defined according to the following conditions. If

1. $t\,[\underline{x}]\,, t'\,[\underline{x}]\,, t''\,[\underline{x}]\,, s\,[\underline{x}, y]\,, s'\,[\underline{x}, y]\,, r\,[\underline{x}, y, z]\,, q\,[\underline{x}, y, z, u]$ are terms in context;
2. $\phi\,[\underline{x}]\,, \phi'\,[\underline{x}]\,, \psi\,[\underline{x}, y]$ are small propositions in context;
3. $A\,[\underline{x}]\,, A'\,[\underline{x}]\,, B\,[\underline{x}, y]$ are sets in context;
4. $\eta\,[\underline{x}]\,, \eta'\,[\underline{x}]\,, \rho\,[\underline{x}, y]$ are propositions in context;
5. $D\,[\underline{x}]\,, E\,[\underline{x}, y]$ are collections in context,

then

1. $x_i\,[\underline{x}]$ is a term in context;
the empty set eliminator $\mathsf{emp}_0(t)\,[\underline{x}]$ is a term in context;
the singleton constant $\star\,[\underline{x}]$ and *the singleton eliminator* $El_{N_1}(t,t')\,[\underline{x}]$ are terms in context;
the zero constant $0\,[\underline{x}]$, *the successor constructor* $\mathsf{succ}(t)\,[\underline{x}]$ and *the eliminator of natural numbers* $El_N(t,t',(y,z)r)\,[\underline{x}]$ are terms in context[1];
the lambda abstraction of dependent product $\lambda y.s\,[\underline{x}]$ and *its application* $\mathsf{Ap}(t,t')\,[\underline{x}]$ are terms in context;
the pairing of strong indexed sum $\langle t,t'\rangle\,[\underline{x}]$ and its eliminator $El_\Sigma(t,(y,z)r)\,[\underline{x}]$ are terms in context;
the first injection of binary disjoint sum $\mathsf{inl}(t)\,[\underline{x}]$, *its second injection* $\mathsf{inr}(t)\,[\underline{x}]$ and *its eliminator* $El_+(t,(y)s,(y)s')\,[\underline{x}]$ are terms in context;
the empty list $\epsilon\,[\underline{x}]$, *the list constructor* $\mathsf{cons}(t,t')\,[\underline{x}]$ and *its eliminator* $El_{List}(t,t',(y,z,u)q)\,[\underline{x}]$ are terms in context;
the false eliminator $\mathsf{r}_0(t)\,[\underline{x}]$ is a term in context;
the pairing of conjunction $\langle t,_\wedge t'\rangle\,[\underline{x}]$, and *its first and second projections* $\pi_1^\wedge(t)\,[\underline{x}]$ and $\pi_2^\wedge(t)\,[\underline{x}]$ are terms in context;
the first injection of disjunction $\mathsf{inl}_\vee(t)\,[\underline{x}]$, *the second injection of disjunction* $\mathsf{inr}_\vee(t)\,[\underline{x}]$ and *its eliminator* $El_\vee(t,(y)s,(y)s')\,[\underline{x}]$ are terms in context;
the lambda abstraction of implication $\lambda_\rightarrow y.s\,[\underline{x}]$ and *its application* $\mathsf{Ap}_\rightarrow(t,t')\,[\underline{x}]$ are terms in context;
the pairing of existential quantification $\langle t,_\exists t'\rangle\,[\underline{x}]$ and *its eliminator* $El_\exists(t,(y,z)r)\,[\underline{x}]$ are terms in context;
the lambda abstraction of universal quantification $\lambda_\forall y.s\,[\underline{x}]$ and *its application* $\mathsf{Ap}_\forall(t,t')\,[\underline{x}]$ are terms in context;
the Propositional Identity term constructor $\mathsf{id}(t)\,[\underline{x}]$ and *its eliminator* $El_{\mathsf{Id}}(t,t',t'',(y)s)\,[\underline{x}]$ [2] are terms in context;
the empty set code $\widehat{N_0}[\underline{x}]$, *the singleton code* $\widehat{N_1}[\underline{x}]$, *the natural numbers set code* $\widehat{N}[\underline{x}]$, *the dependent product code* $\widehat{(\Pi y\in A)}s[\underline{x}]$, *the dependent sum code* $\widehat{(\Sigma y\in A)}s[\underline{x}]$, *the disjoint sum code* $t\widehat{+}t'[\underline{x}]$, *the list code* $\widehat{List}(t)[\underline{x}]$, *the falsum code* $\widehat{\bot}$, *the conjunction code* $t\widehat{\wedge}t'$, *the disjunction code* $t\widehat{\vee}t'$, *the implication code* $t\widehat{\rightarrow}t'$, *the existential quantification code* $\widehat{(\exists y\in A)}s\,[\underline{x}]$, *the universal quantification code* $\widehat{(\forall y\in A)}s\,[\underline{x}]$ and *the propositional identity code* $\widehat{\mathsf{Id}}(A,t,t')\,[\underline{x}]$ are terms in context;

2. $\bot\,[\underline{x}]$ and $\tau(t)\,[\underline{x}]$ are small propositions in context;
$\phi\wedge\phi'\,[\underline{x}]$, $\phi\vee\phi'\,[\underline{x}]$ and $\phi\rightarrow\phi'\,[\underline{x}]$ are small propositions in context;
$(\exists y\in A)\,\psi\,[\underline{x}]$, $(\forall y\in A)\,\psi\,[\underline{x}]$ and $\mathsf{Id}(A,t,t')\,[\underline{x}]$ are small propositions in context;

3. $\phi\,[\underline{x}]$ is a set in context;
$N_0\,[\underline{x}]\,, N_1\,[\underline{x}]$ and $N\,[\underline{x}]$ are sets in context;
$(\Pi y\in A)\,B\,[\underline{x}]$, $(\Sigma y\in A)\,B\,[\underline{x}]$, $A+A'\,[\underline{x}]$ and $List(A)\,[\underline{x}]$ are sets in context;

4. $\phi\,[\underline{x}]$ is a proposition in context;
$\eta\wedge\eta'\,[\underline{x}]$, $\eta\vee\eta'\,[\underline{x}]$ and $\eta\rightarrow\eta'\,[\underline{x}]$ are propositions in context;

[1] The rules for these constructors derive from those of $List(N_1)$ in **mTT** by identifying 0 with ϵ, $\mathsf{succ}(t)$ with $\mathsf{cons}(t,\star)$ and $El_N(t,t',(y,z)r)$ with $El_{List(N_1)}(t,t',(y,y',z)r)$.

[2] In the rules for $\mathsf{Id}(A,a,b)$ of **mTT** the eliminator $El_{\mathsf{Id}}(p,(x)c)$ is substituted by an eliminator $El_{\mathsf{Id}}(a,b,p,(x)c)$ with explicit reference to $a\in A$ and $b\in A$. The rules remain the same.

$(\exists y \in D)\,\rho\,[\underline{x}]$ and $(\forall y \in D)\,\rho\,[\underline{x}]$ and $\mathsf{Id}(D,t,t')\,[\underline{x}]$ are propositions in context;

5. $\eta\,[\underline{x}]$ and $A\,[\underline{x}]$ are collections in context;
   $\mathsf{Set}\,[\underline{x}]$ is a collection in context;
   $\mathsf{props}_{\mathsf{s}}\,[\underline{x}]$ and $A \to \mathsf{props}_{\mathsf{s}}\,[\underline{x}]$ are collections in context;
   $(\Sigma y \in D)E\,[\underline{x}]$ is a collection in context.

For sets in context $A\,[\underline{x}]$ we define an abbreviation $\widehat{A}\,[\underline{x}]$ as follows:

1. $\widehat{\bot}$, $\widehat{N_0}$, $\widehat{N_1}$ and $\widehat{N}$ were already defined;
2. $\widehat{((\Pi y \in A)\,B)} = \widehat{(\Pi y \in A)}\,\widehat{B}$, $\widehat{((\Sigma y \in A)\,B)} = \widehat{(\Sigma y \in A)}\,\widehat{B}$,
3. $\widehat{A + A'} = \widehat{A}\widehat{+}\widehat{A'}$, $\widehat{List(A)} = \widehat{List}(\widehat{A})$,
4. $\widehat{\phi \wedge \phi'} = \widehat{\phi}\,\widehat{\wedge}\,\widehat{\phi'}$, $\widehat{\phi \vee \phi'} = \widehat{\phi}\,\widehat{\vee}\,\widehat{\phi'}$, $\widehat{\phi \to \phi'} = \widehat{\phi}\,\widehat{\to}\,\widehat{\phi'}$,
5. $\widehat{((\exists y \in A)\,\psi)} = \widehat{(\exists y \in A)}\,\widehat{\psi}$, $\widehat{((\forall y \in A)\,\psi)} = \widehat{(\forall y \in A)}\,\widehat{\psi}$, $\widehat{\mathsf{Id}(A,t,s)} = \widehat{\mathsf{Id}}\,(A,t,s)$,
6. $\widehat{\tau(t)} = t$.

It is clear that the previous definition is overabundant with respect to the common use in type theory. We introduced some terms which we will never find in any standard type theory, as for example the term $0\widehat{\wedge}El_{N_1}(\lambda x.x, \lambda_{\to} y.y)$ which is obtained by gluing together terms which usually have types which are not compatible. For example 0 is usually typed as a natural number, while $\widehat{\wedge}$ connects codes for small propositions.

3 The realizability interpretation for $\mathbf{mTT}^s$

3.1 The system $\widehat{ID_1}$

The preliminary step in the presentation of the Kleene realizability interpretation consists in presenting the theory of *Inductive Definitions* $\widehat{ID_1}$ in which we will interpret $\mathbf{mTT}^s$. The system $\widehat{ID_1}$ is a predicative fragment of second-order arithmetic, more precisely it is the predicative fragment of second-order arithmetic extending Peano arithmetics with some (not necessarily least) fix points for each positive arithmetical operator. Its number terms are number variables (we assume that these variables are equal to those of $\mathbf{mTT}^s$), the constant 0 and the terms built by applying the unary successor functional symbol *succ* and the binary sum and product functional symbols $+$ and $*$ to number terms. Set terms are only set variables $X, Y, Z....$ The *arithmetical* formulas are obtained starting from $t = s$ and $t\varepsilon X$ with t, s number terms and X a set variable, by applying the connectives $\wedge, \vee, \neg, \to$ and the number quantifiers $\forall x$, $\exists x$. Moreover let us give the following two definitions.

▶ **Definition 2.** An occurrence of a set variable X in an arithmetical formula φ is *positive* or *negative* according to the following conditions:
1. it is positive if φ is $t\varepsilon X$ for some number term t,
2. it is positive (negative) if φ is $\psi \wedge \psi'$, $\psi' \wedge \psi$, $\psi \vee \psi'$, $\psi' \vee \psi$, $\psi' \to \psi$, $\exists x\,\psi$ or $\forall x\,\psi$ and it is a positive (negative) occurrence of X in ψ or φ is $\psi \to \psi'$ or $\neg\psi$ and the occurrence of X is a negative (positive) occurrence of X in ψ.

▶ **Definition 3.** An arithmetical formula φ with exactly one free number variable x and one free set variable X which occurs only positively is called an *admissible* formula.
In order to define the system $\widehat{ID_1}$ we add to the language of arithmetic a unary predicate symbol P_φ for every admissible formula φ . The atomic formulas of $\widehat{ID_1}$ are

1. $t = s$ with t and s number terms,
2. $t \varepsilon X$ with t a number term and X a set variable,
3. $P_\varphi(t)$ with t a number term and φ an admissible formula.

All formulas of $\widehat{ID_1}$ are obtained by atomic formulas by applying connectives, number quantifiers and set quantifiers.

The axioms of $\widehat{ID_1}$ are the axioms of Peano Arithmetic plus the following three axiom schemata:

1. *Comprehension schema*: for all formulas $\varphi(x)$ of $\widehat{ID_1}$ without set quantifiers
$$\exists X \forall x (x \varepsilon X \leftrightarrow \varphi(x))$$
2. *Induction schema*: for all formulas $\varphi(x)$ of $\widehat{ID_1}$ without set quantifiers
$$(\varphi(0) \wedge \forall x(\varphi(x) \rightarrow \varphi(succ(x)))) \rightarrow \forall x\, \varphi(x)$$
3. *Fix point schema*: for all admissible formulas φ
$$\varphi[P_\varphi/X] \leftrightarrow P_\varphi(x)$$
where $\varphi[P_\varphi/X]$ is the result of substituting in φ every atomic formula $t\varepsilon X$ with $P_\varphi(t)$.

The system $\widehat{ID_1}$ allows us to define predicates as fix points, by using axiom schema 3, if they are presented in a appropriate way (i. e. using admissible formulas).

A *definable class* $\mathcal{C}$ of $\widehat{ID_1}$ is a formal writing $\{x|\varphi(x)\}$ where $\varphi(x)$ is a formula of $\widehat{ID_1}$. In this case we write $x\varepsilon\mathcal{C}$ as a shorthand for $\varphi(x)$.

Notation of computable operators in $\widehat{ID_1}$.

As it is well known, it is certainly possible to express a Gödelian coding of recursive functions in $\widehat{ID_1}$ using Kleene's predicate since it is already possible to do this in PA. In particular we can consider a definitional extension of $\widehat{ID_1}$ (which we still call $\widehat{ID_1}$) in which there are first-order terms with Kleene's brackets $\{t\}(s)$ and there is a predicate $\{t\}(s)\downarrow$ stating that the term with Kleene's brackets is well defined (s is in the domain of the recursive function coded by t). We will write $\{t\}(s_1, ..., s_n)$ as a shorthand defined by induction: it is $\{t\}(s_1)$ if $n = 1$ while if $n > 1$ and if we have already defined $\{t\}(s_1, ..., s_n)$, then $\{t\}(s_1, ..., s_{n+1}) = \{\{t\}(s_1, ..., s_n)\}(s_{n+1})$. We denote by $\mathbf{succ}$ a numeral for which $\{\mathbf{succ}\}(x) = succ(x)$ in $\widehat{ID_1}$.

As we well know, the s-m-n lemma (see e. g. [18]) gives the structure of a partial combinatorial algebra to natural numbers endowed with Kleene application and this structure can be expressed in $\widehat{ID_1}$. In particular we can find numerals $\mathbf{p}, \mathbf{p}_1, \mathbf{p}_2$ representing a fixed primitive recursive bijective pairing function with primitive recursive first and second projections. We will write $p_1(x)$, $p_2(x)$ and $\langle x, y\rangle$ as abbreviations for $\{\mathbf{p}_1\}(x)$, $\{\mathbf{p}_2\}(x)$ and $\{\mathbf{p}\}(x, y)$ respectively. It is also possible to define a numeral **ite**[3] representing the definition by cases ($\{\mathbf{ite}\}(n, m, l) \simeq$ [4]m if $n = 0$, $\{\mathbf{ite}\}(n, m, l) \simeq l$ if $n \neq 0$). We can also encode recursively finite list of natural numbers with natural numbers in such a way that the empty list is coded by 0 and the concatenation is a recursive function which can be coded by a numeral **cnc**. We have moreover numerals **rec** and **listrec** representing natural numbers recursion and lists recursion. These numbers in particular satisfy the following requirements:

[3] if then else
[4] $a \simeq b$ means that $a\downarrow$ if and only if $b\downarrow$ and in this case $a = b$ in $\widehat{ID_1}$.

1. $\{\mathbf{rec}\}(n, m, 0) \simeq n$;
2. $\{\mathbf{rec}\}(n, m, k+1) \simeq \{m\}(k, \{\mathbf{rec}\}(n, m, k))$;
3. $\{\mathbf{listrec}\}(n, m, 0) \simeq n$;
4. $\{\mathbf{listrec}\}(n, m, \mathbf{cnc}(k, l)) \simeq \{m\}(k, l, \{\mathbf{listrec}\}(n, m, k))$.

For this representation of lists, the component functions $(-)_j$, turn out to be recursive.

Moreover we can always define λ-terms $\Lambda x.t$ in $\widehat{ID_1}$ for terms t built with numerals, variables and Kleene application, in such a way that $\{\Lambda x.t\}(n) \simeq t[n/x]$ and it holds that $\{\Lambda x_1 ... \Lambda x_{\mathsf{n}}.t\}(n) \simeq \Lambda x_2 ... \Lambda x_{\mathsf{n}}.t[n/x_1]$; moreover if all variables of t are among $x_1, ..., x_{\mathsf{k}}$, then there is a numeral $\mathbf{n}$ for which $\widehat{ID_1} \vdash \Lambda x_1 ... \Lambda x_{\mathsf{k}}.t = \mathbf{n}$.

3.2 The definition of interpretation

The realizability interpretation for $\mathbf{mTT}^s$ we are going to describe is a modification of Beeson's realizability semantics [2] for the extensional version of first order Martin-Löf's type theory with one universe [16]. So it will be given in $\widehat{ID_1}$ as Beeson's one. Here we describe the key points of such an interpretation on which we follow Beeson's semantics:

- all types of $\mathbf{mTT}^s$ are interpreted as quotients of definable classes of $\widehat{ID_1}$, intended as classes of "their realizers". In particular we use Beeson's technique of interpreting Martin-Löf's universe to interpret the collection of (codes for) small propositions of $\mathbf{mTT}^s$. In order to do this it is crucial to have fix points and hence this is why we work in the theory $\widehat{ID_1}$;
- terms are interpreted as (codes) of recursive functions;
- equality between terms in context is interpreted as extensional equality of recursive functions;
- the interpretation of substitution will be proven to be equivalent to the substitution in interpretation;
- we interpret λ-abstraction by using *s-m-n* lemma of computability, but then, in order to validate the condition of the previous point, we impose equality of type-theoretic functions to be extensional. Therefore the principle of Extensional Equality of Functions will turn out to be valid in our model.

Instead we do not follow Beeson's semantics in the interpretation of propositions:

- in order to validate **formal Church Thesis** we interpret propositions as trivial[5] quotients of original Kleene realizability. As a consequence Martin-Löf's isomorphism of propositions-as-sets together with the validity of the **Axiom of Choice** is not validated in our realizability semantics contrary to Beeson's one.

We can summarize the interpretation of terms and types with the following table:

Terms	(codes) of recursive functions
Collections	Quotients of definable classes $(C, \simeq)$
Propositions	quotients of definable classes on trivial $\simeq$

[5] A quotient is trivial if it is determined by a trivial relation i. e. a relation for which all pairs of elements are equivalent.

The interpretation of terms

Before giving the interpretation of $\mathbf{mTT}^s$-terms, we need to present explicitly a convention about how to encode $\mathbf{mTT}^s$-sets with numerals. We will code sets as $\{\mathbf{p}\}(a, \langle b_1, ..., b_{\mathsf{n}}\rangle)$, where a is a number coding a particular constructor and $\langle b_1, ..., b_{\mathsf{n}}\rangle$ is a lists of codes for ingredients needed by the constructor itself. The following table makes evident the choices for a:

N_0, N_1, N	Π	Σ	$+$	$List$	$\bot$	$\wedge$	$\vee$	$\rightarrow$	$\exists$	$\forall$	Id
1	2	3	4	5	6	7	8	9	10	11	12

Notice that codes for small propositions must have $a > 5$.

We can now proceed to the definition of the interpretation of $\mathbf{mTT}^s$-terms.

▶ **Definition 4.** Terms in context $t[x_1, ..., x_{\mathsf{n}}]$ are interpreted as

$$\mathcal{I}(t[x_1, ..., x_{\mathsf{n}}]) = \Lambda x_1 ... \Lambda x_{\mathsf{n}}.\mathcal{I}(t)$$

where $\mathcal{I}(t)$ are terms of the extended language of $\widehat{ID_1}$ defined as follows

1. If x is a variable, then $\mathcal{I}(x) = x$;
2. $\mathcal{I}(\mathsf{emp}_0(t)) = \mathcal{I}(\mathsf{r}_0) = 0$;
3. $\mathcal{I}(\star) = 0$ and $\mathcal{I}(El_{N_1}(t, t')) = \mathcal{I}(t')$;
4. $\mathcal{I}(0) = 0$ and $\mathcal{I}(\mathsf{succ}(t)) = \{\mathbf{succ}\}(\mathcal{I}(t))$,
   $\mathcal{I}(El_N(t, t', (y, z)r)) = \{\mathbf{rec}\}(\mathcal{I}(t'), \Lambda y.\Lambda z.\mathcal{I}(r), \mathcal{I}(t))$;
5. $\mathcal{I}(\lambda y.s) = \mathcal{I}(\lambda_{\rightarrow} y.s) = \mathcal{I}(\lambda_{\forall} y.s) = \Lambda y.\mathcal{I}(s)$,
   $\mathcal{I}(\mathsf{Ap}(t, t')) = \mathcal{I}(\mathsf{Ap}_{\rightarrow}(t, t')) = \mathcal{I}(\mathsf{Ap}_{\forall}(t, t')) = \{\mathcal{I}(t)\}(\mathcal{I}(t'))$;
6. $\mathcal{I}(\langle t, t'\rangle) = \mathcal{I}(\langle t,_{\wedge} t'\rangle) = \mathcal{I}(\langle t,_{\exists} t'\rangle) = \{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t'))$,
   $\mathcal{I}(El_{\Sigma}(t, (y, z)r)) = \mathcal{I}(El_{\exists}(t, (y, z)r)) = \{\Lambda y.\Lambda z.\mathcal{I}(r)\}(\{\mathbf{p}_1\}(\mathcal{I}(t)), \{\mathbf{p}_2\}(\mathcal{I}(t)))$,
   $\mathcal{I}(\pi_1^{\wedge}(t)) = \{\mathbf{p}_1\}(\mathcal{I}(t))$,
   $\mathcal{I}(\pi_2^{\wedge}(t)) = \{\mathbf{p}_2\}(\mathcal{I}(t))$;
7. $\mathcal{I}(\mathsf{inl}(t)) = \mathcal{I}(\mathsf{inl}_{\vee}(t)) = \{\mathbf{p}\}(0, \mathcal{I}(t))$,
   $\mathcal{I}(\mathsf{inr}(t)) = \mathcal{I}(\mathsf{inr}_{\vee}(t)) = \{\mathbf{p}\}(1, \mathcal{I}(t))$,
   $\mathcal{I}(El_{+}(t, (y)s, (y)s')) = \mathcal{I}(El_{\vee}(t, (y)s, (y)s')) =$
   $\{\mathbf{ite}\}(\mathbf{p}_1(\mathcal{I}(t)), \{\Lambda y.\mathcal{I}(s)\}(\{\mathbf{p}_2\}(\mathcal{I}(t))), \{\Lambda y.\mathcal{I}(s')\}(\{\mathbf{p}_2\}(\mathcal{I}(t))))$;
8. $\mathcal{I}(\epsilon) = 0$ and $\mathcal{I}(\mathsf{cons}(t, t')) = \{\mathbf{cnc}\}(\mathcal{I}(t), \mathcal{I}(t'))$,
   $\mathcal{I}(El_{List}(t, t', (y, z, u)q)) = \{\mathbf{listrec}\}(\mathcal{I}(t'), \Lambda y.\Lambda z.\Lambda u.\mathcal{I}(q), \mathcal{I}(t))$;
9. $\mathcal{I}(\mathsf{id}(t)) = 0$,
   $\mathcal{I}(El_{\mathsf{Id}}(t, t', t'', (y)s)) = \{\Lambda y.\mathcal{I}(s)\}(\mathcal{I}(t))$;
10. $\mathcal{I}(\widehat{N_0}) = \{\mathbf{p}\}(1, 0)$, $\mathcal{I}(\widehat{N_1}) = \{\mathbf{p}\}(1, 1)$ and $\mathcal{I}(\widehat{N}) = \{\mathbf{p}\}(1, 2)$,
    $\mathcal{I}((\widehat{\Pi y \in A})s) = \{\mathbf{p}\}(2, (\{\mathbf{p}\}(\mathcal{I}(\widehat{A}), (\Lambda y.\mathcal{I}(s)))))$,
    $\mathcal{I}((\widehat{\Sigma y \in A})s) = \{\mathbf{p}\}(3, (\{\mathbf{p}\}(\mathcal{I}(\widehat{A}), (\Lambda y.\mathcal{I}(s)))))$,
    $\mathcal{I}(t \widehat{+} t') = \{\mathbf{p}\}(4, (\{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t')))$,
    $\mathcal{I}(\widehat{List}(t)) = \{\mathbf{p}\}(5, \mathcal{I}(t))$,
    $\mathcal{I}(\widehat{\bot}) = \{\mathbf{p}\}(6, 0)$,
    $\mathcal{I}(t \widehat{\wedge} t') = \{\mathbf{p}\}(7, (\{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t')))$,
    $\mathcal{I}(t \widehat{\vee} t') = \{\mathbf{p}\}(8, (\{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t')))$,
    $\mathcal{I}(t \widehat{\rightarrow} t') = \{\mathbf{p}\}(9, (\{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t')))$,
    $\mathcal{I}((\widehat{\exists y \in A})s) = \{\mathbf{p}\}(10, (\{\mathbf{p}\}(\mathcal{I}(\widehat{A}), (\Lambda y.\mathcal{I}(s)))))$,
    $\mathcal{I}((\widehat{\forall y \in A})s) = \{\mathbf{p}\}(11, (\{\mathbf{p}\}(\mathcal{I}(\widehat{A}), (\Lambda y.\mathcal{I}(s)))))$,

$$\mathcal{I}(\widehat{\mathsf{Id}}(A,t,t')) = \{\mathbf{p}\}(12, (\{\mathbf{p}\}(\mathcal{I}(\widehat{A}), (\{\mathbf{p}\}(\mathcal{I}(t), \mathcal{I}(t')))))),$$

For the sake of example let us consider the interpretation of the term in context $t[x,y,z]$ defined as $\widehat{\mathsf{Id}}(\mathsf{Id}(N,x,x),y,z)[x,y,z]$:

$$\begin{aligned}
\mathcal{I}(t)[x,y,z]) &= \Lambda x.\Lambda y.\Lambda z.\mathcal{I}(\widehat{\mathsf{Id}}(\mathsf{Id}(N,x,x),y,z)) \\
&= \Lambda x.\Lambda y.\Lambda z.\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\mathcal{I}(\widehat{\mathsf{Id}(N,x,x)}), \{\mathbf{p}\}(y,z))) \\
&= \Lambda x.\Lambda y.\Lambda z.\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\mathcal{I}(\widehat{\mathsf{Id}}(N,x,x)), \{\mathbf{p}\}(y,z))) \\
&= \Lambda x.\Lambda y.\Lambda z.\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\mathcal{I}(\widehat{N}), \{\mathbf{p}\}(x,x))),\ \{\mathbf{p}\}(y,z))) \\
&= \Lambda x.\Lambda y.\Lambda z.\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\{\mathbf{p}\}(12,\ \{\mathbf{p}\}(\{\mathbf{p}\}(1,2), \{\mathbf{p}\}(x,x))),\ \{\mathbf{p}\}(y,z))).
\end{aligned}$$

We say that an interpretation of a term in context $t[\underline{x}]$ is well defined if $\mathcal{I}(t[\underline{x}]) \downarrow$ is provable in $\widehat{ID_1}$. Notice that the interpretations of terms in non-empty contexts are always well defined.

Notice moreover that in $\widehat{ID_1}$

1. $\mathcal{I}(El_{N_1}(\star, t')) \simeq \mathcal{I}(t')$;
2. $\mathcal{I}(El_N(0, t, (y,z)s)) \simeq \mathcal{I}(t)$;
3. $\mathcal{I}(El_N(\mathsf{succ}(t'), t, (y,z)s)) \simeq \mathcal{I}(s)[\mathcal{I}(t')/y, \mathcal{I}(El_N(t', t, (y,z)s))/z]$
4. $\mathcal{I}(\mathsf{Ap}(\lambda y.s, t)) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
5. $\mathcal{I}(\mathsf{Ap}_{\rightarrow}(\lambda_{\rightarrow} y.s, t)) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
6. $\mathcal{I}(\mathsf{Ap}_{\forall}(\lambda_{\forall} y.s, t)) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
7. $\mathcal{I}(El_{\Sigma}(\langle t, t'\rangle, (y,z)r)) \simeq \mathcal{I}(r)[\mathcal{I}(t)/y, \mathcal{I}(t')/z]$;
8. $\mathcal{I}(El_{\exists}(\langle t,_{\exists} t'\rangle, (y,z)r)) \simeq \mathcal{I}(r)[\mathcal{I}(t)/y, \mathcal{I}(t')/z]$;
9. $\mathcal{I}(\pi_1^{\wedge}(\langle t,_{\wedge} t'\rangle)) \simeq \mathcal{I}(t)$;
10. $\mathcal{I}(\pi_2^{\wedge}(\langle t,_{\wedge} t'\rangle)) \simeq \mathcal{I}(t')$;
11. $\mathcal{I}(El_{+}(\mathsf{inl}(t), (y)s, (y)s')) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
12. $\mathcal{I}(El_{+}(\mathsf{inr}(t), (y)s, (y)s')) \simeq \mathcal{I}(s')[\mathcal{I}(t)/y]$;
13. $\mathcal{I}(El_{\vee}(\mathsf{inl}_{\vee}(t), (y)s, (y)s')) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
14. $\mathcal{I}(El_{\vee}(\mathsf{inr}_{\vee}(t), (y)s, (y)s')) \simeq \mathcal{I}(s')[\mathcal{I}(t)/y]$;
15. $\mathcal{I}(El_{\mathsf{Id}}(t, t, \mathsf{id}(t), (y)s)) \simeq \mathcal{I}(s)[\mathcal{I}(t)/y]$;
16. $\mathcal{I}(El_{List}(\epsilon, t', (y,z,u)q)) \simeq \mathcal{I}(t')$;
17. $\mathcal{I}(El_{List}(\mathsf{cons}(t, t''), t', (y,z,u)q)) \simeq \mathcal{I}(q)[\mathcal{I}(t)/y, \mathcal{I}(t'')/z, \mathcal{I}(El_{List}(t, t', (y,z,u)q))/u]$.

The interpretation of sets

Here we define the interpretation of sets in $\mathbf{mTT}^s$ with the exception of those obtained using $\tau(p)$ for some term p. Every such a set is interpreted as a definable quotient of a definable class of $\widehat{ID_1}$ (and actually of HA). This means that every set A is interpreted as a pair

$$\mathcal{I}(A) = (\ \mathcal{J}(A)\ ,\ \sim_{\mathcal{I}(A)}\ )$$

where $\mathcal{J}(A)$ is a definable class of $\widehat{ID_1}$ and $\sim_{\mathcal{I}(A)}$ is a definable equivalence relation on the class $\mathcal{J}(A)$.

Since sets in $\mathbf{mTT}$ include small propositions, here we also define a realizability relation between natural numbers and propositions. Indeed it is more convenient to define the realizability interpretation of propositions by adopting an extension of usual Kleene's interpretation of intuitionistic connectives.

Note that we use the notation $\mathcal{I}(A)[s/y]$ to mean the definable class in which we substitute y with s in the membership and in the equivalence relation of $\mathcal{I}(A)$.

▶ **Definition 5.** We define in $\widehat{ID_1}$ a realizability relation $n \Vdash \phi$ between natural numbers and small propositions, by induction on the definition of small propositions ϕ, simultaneously together with the definition of the following formulas $n\varepsilon\mathcal{J}(A)$ and $n \sim_{\mathcal{I}(A)} m$ for sets A, by induction on the definition of sets (with the exception of those obtained using $\tau(p)$ for some term p), as follows:

($\bot$) $n \Vdash \bot$ is $\bot$;
($\wedge$) $n \Vdash \phi \wedge \phi'$ is $(p_1(n) \Vdash \phi) \wedge (p_2(n) \Vdash \phi')$;
($\vee$) $n \Vdash \phi \vee \phi'$ is $(p_1(n) = 0 \wedge p_2(n) \Vdash \phi) \vee (p_1(n) \neq 0 \wedge p_2(n) \Vdash \phi')$;
($\rightarrow$) $n \Vdash \phi \rightarrow \phi'$ is $\forall t\,((t \Vdash \phi) \rightarrow (\{n\}(t) \Vdash \phi'))$;
($\exists$) $n \Vdash (\exists x \in A)\,\psi$ is $p_1(n)\,\varepsilon\,\mathcal{J}(A) \wedge (p_2(n) \Vdash \psi)[p_1(n)/x]$;
($\forall$) $n \Vdash (\forall x \in A)\,\psi$ is $\forall x\,(x\,\varepsilon\,\mathcal{J}(A) \rightarrow (\{n\}(x) \Vdash \psi))$;
(Id) $n \Vdash \mathsf{Id}(A,t,s)$ is $\mathcal{I}(t) \sim_{\mathcal{I}(A)} \mathcal{I}(s)$;
(N_0) $n\,\varepsilon\,\mathcal{J}(N_0)$ is $\bot$ and
$n \sim_{\mathcal{I}(N_0)} m$ is $\bot$;
(N_1) $n\,\varepsilon\,\mathcal{J}(N_1)$ is $n = 0$ and
$n \sim_{\mathcal{I}(N_1)} m$ is $n = 0 \wedge n = m$;
(N) $n\,\varepsilon\,\mathcal{J}(N)$ is $n = n$ and
$n \sim_{\mathcal{I}(N)} m$ is $n = m$;
(Π) $n\,\varepsilon\,\mathcal{J}((\Pi x \in A)\,B)$ is
$\forall x\,(x\,\varepsilon\,\mathcal{J}(A) \rightarrow \{n\}(x) \in \mathcal{J}(B)) \wedge \forall x \forall y\,(x \sim_{\mathcal{I}(A)} y \rightarrow \{n\}(x) \sim_{\mathcal{I}(B)} \{n\}(y))$[6] and
$n \sim_{\mathcal{I}((\Pi x \in A)\,B)} m$ is
$n\,\varepsilon\,\mathcal{J}((\Pi x \in A)B) \wedge m\,\varepsilon\,\mathcal{J}((\Pi x \in A)B) \wedge \forall x\,(x\,\varepsilon\,\mathcal{J}(A) \rightarrow \{n\}(x) \sim_{\mathcal{I}(B)} \{m\}(x))$;
(Σ) $n\,\varepsilon\,\mathcal{J}((\Sigma x \in A)\,B)$ is $p_1(n)\,\varepsilon\,\mathcal{J}(A) \wedge \forall x\,(x \sim_{\mathcal{I}(\mathsf{A})} p_1(n) \rightarrow p_2(n)\,\varepsilon\,\mathcal{J}(B))$ and
$n \sim_{\mathcal{I}((\Sigma x \in A)\,B)} m$ is the conjunction of $n\,\varepsilon\,\mathcal{J}((\Sigma x \in A)B) \wedge m\,\varepsilon\,\mathcal{J}((\Sigma x \in A)B)$ and
$p_1(n) \sim_{\mathcal{I}(A)} p_1(m) \wedge \forall x\,(x \sim_{\mathcal{I}(A)} p_1(n) \rightarrow p_2(n) \sim_{\mathcal{I}(B)} p_2(m))$;
($+$) $n\,\varepsilon\,\mathcal{J}(A + A')$ is $(p_1(n) = 0 \wedge p_2(n)\,\varepsilon\,\mathcal{J}(A)) \vee (p_1(n) = 1 \wedge p_2(n)\,\varepsilon\,\mathcal{J}(A'))$ and
$n \sim_{\mathcal{I}(A+A')} m$ is the conjunction of $n\,\varepsilon\,\mathcal{J}(A + A') \wedge m\,\varepsilon\,\mathcal{J}(A + A') \wedge p_1(n) = p_1(m)$ and
$(p_1(n) = 0 \wedge p_2(n) \sim_{\mathcal{I}(A)} p_2(m)) \vee (p_1(n) = 1 \wedge p_2(n) \sim_{\mathcal{I}(A')} p_2(m))$;
($List$) $n\,\varepsilon\,\mathcal{J}(List(A))$ is $\forall j\,(j < lh(n) \rightarrow (n)_j\,\varepsilon\,\mathcal{J}(A))$ and
$n \sim_{\mathcal{I}(List(A))} m$ is the conjunction of $n\,\varepsilon\,\mathcal{J}(List(A)) \wedge m\,\varepsilon\,\mathcal{J}(List(A))$ and
$lh(n) = lh(m) \wedge \forall j\,(j < lh(n) \rightarrow (n)_j \sim_{\mathcal{I}(A)} (m)_j)$;
(ψ) $n\,\varepsilon\,\mathcal{J}(\psi)$ is $n \Vdash \psi$ and
$n \sim_{\mathcal{I}(\psi)} m$ is $n\,\varepsilon\,\mathcal{J}(\psi) \wedge m\,\varepsilon\,\mathcal{J}(\psi)$ (i. e. *proof-irrelevance*).

▶ Remark. We can notice some preliminary properties of this realizability interpretation:

1. for every set A we have that $\sim_{\mathcal{I}(A)}$ is really a definable equivalence relation on the definable class $\mathcal{J}(A)$, in fact

$$n\,\varepsilon\,\mathcal{J}(A) \vdash_{\widehat{ID_1}} n \sim_{\mathcal{I}(A)} n$$

$$n \sim_{\mathcal{I}(A)} m \vdash_{\widehat{ID_1}} m \sim_{\mathcal{I}(A)} n$$

$$n \sim_{\mathcal{I}(A)} m \wedge m \sim_{\mathcal{I}(A)} l \vdash_{\widehat{ID_1}} n \sim_{\mathcal{I}(A)} l$$

[6] Note that the variable x may be in $\mathcal{I}(B)$ here and in the following definition for Π and Σ sets, as it comes from the definition of the untyped syntax.

2. for every set A we have that

$$n \sim_{\mathcal{I}(A)} m \vdash_{\widehat{ID_1}} n \,\varepsilon\, \mathcal{J}(A) \wedge m \,\varepsilon\, \mathcal{J}(A)$$

3. if *numerical* sets are defined according to the following conditions
   a. N_0, N_1 and N are numerical sets;
   b. if A and B are numerical sets, then $(\Sigma x \in A)\, B$, $A + B$ and $List(A)$ (if they are well defined) are numerical sets,

   then the equality of the interpretation of numerical sets is numerical, which means that

$$n \sim_{\mathcal{I}(A)} m \vdash_{\widehat{ID_1}} n = m$$

4. for all small propositions ψ, the equivalence relation $\sim_{\mathcal{I}(\psi)}$ is trivial (i. e. all pairs of elements of $\mathcal{I}(\psi)$ are equivalent). This means that uniqueness of propositional proofs, called *proof-irrelevance*, is imposed.

The encoding of all $\mathbf{mTT}^s$-sets

In the previous sections we have seen the interpretation of $\mathbf{mTT}^s$-sets which include small propositions. It remains to define the interpretation of proper collections, including that of sets, small propositions and small propositional functions on a set.

The interpretation of the collection of small propositions Set in $\widehat{ID_1}$ is the most difficult point and to define it we mimick the technique adopted by Beeson [2] to interpret Martin-Löf's universe via a fix point of some arithmetical operator with positive parameters. Hence, it is to define the interpretation of Set, and in turn of the collection of small propositions $\mathsf{prop_s}$ and of small propositional functions $A \to \mathsf{prop_s}$ on a set A, that we need to employ the full power of $\widehat{ID_1}$ with fix points.

The idea is to define a $\widehat{ID_1}$-formula which defines codes of sets with their interpretation *as a fix point*. It appears necessary to define a formula called $\mathbf{Set}(n)$ expressing that n is a code of an $\mathbf{mTT}^s$-set together with its realizability interpretation in $\widehat{ID_1}$. As in Beeson's semantics, to define the formula $\mathbf{Set}(n)$ of set codes with their arithmetical interpretation in $\widehat{ID_1}$ we need to encode membership and equality of sets: $t \,\overline{\varepsilon}\, n$ and $t \equiv_n s$. In turn in order to define them, we need to represent the notion of *a family of sets* used to interpret an $\mathbf{mTT}^s$-dependent set. *A family of sets* coded by m on a set coded by n could be described by the formula

$$\mathbf{Set}(n) \wedge \forall t\, (t \,\overline{\varepsilon}\, n \to \mathbf{Set}(\{m\}(t))) \wedge$$

$$\forall t \forall s\, (t \equiv_n s \to (\forall j\, (j \,\overline{\varepsilon}\, \{m\}(t) \leftrightarrow j \,\overline{\varepsilon}\, \{m\}(s)) \wedge \forall j \forall k\, (j \equiv_{\{m\}(t)} k \leftrightarrow j \equiv_{\{m\}(s)} k))).$$

But in this formula not all occurrences of $t \,\overline{\varepsilon}\, n$ and $t \equiv_n s$ are positive. However it is classically equivalent to the conjunction of the formula $\mathbf{Set}(n) \wedge \forall t\, (\neg t \,\overline{\varepsilon}\, n \vee \mathbf{Set}(\{m\}(t)))$ and the formula $\forall t \forall s\, (\neg t \equiv_n s \vee (P_1 \wedge P_2))$ where P_1 is

$$\forall j\, ((\neg j \,\overline{\varepsilon}\, \{m\}(t) \vee j \,\overline{\varepsilon}\, \{m\}(s)) \wedge (\neg j \,\overline{\varepsilon}\, \{m\}(s) \vee j \,\overline{\varepsilon}\, \{m\}(t)))$$

and P_2 is

$$\forall j \forall k\, ((\neg j \equiv_{\{m\}(t)} k \vee j \equiv_{\{m\}(s)} k) \wedge (\neg j \equiv_{\{m\}(s)} k \vee j \equiv_{\{m\}(t)} k))$$

simply substituting all the instances of the schema $a \to b$ with the classically equivalent $\neg a \vee b$. Now the trick consists in defining some predicates $t \,\overline{\not\varepsilon}\, n$ and $t \not\equiv_n s$ mimicking the negations of $t\,\overline{\varepsilon}\,n$ and $t\ \equiv_n\ s$ as fix point predicates, too, in order to get a a **positive** arithmetical operator. Note that the use of a **classical** arithmetic theory with fix points seems unavoidable to be able to interpret the collection of sets via a positive arithmetical operator. From now on we write

$$\mathbf{Fam}(m,n)\ \equiv\ \mathbf{Set}(n)\wedge\forall t\,(t\,\overline{\not\varepsilon}\,n\vee\mathbf{Set}(\{m\}(t)))\wedge\forall t\forall s\,(t\not\equiv_n s\vee(P_1'\wedge P_2'))$$

where P_1' and P_2' are obtained from P_1 and P_2 by substituting negated instances of membership and of equality predicates with their mentioned primitive negated versions

$$P_1'\equiv\forall j\,((j\,\overline{\not\varepsilon}\,\{m\}(t)\vee j\,\overline{\varepsilon}\,\{m\}(s))\wedge(j\,\overline{\not\varepsilon}\,\{m\}(s)\vee j\,\overline{\varepsilon}\,\{m\}(t)))$$

$$P_2'\equiv\forall j\forall k\,((j\not\equiv_{\{m\}(t)}k\vee j\equiv_{\{m\}(s)}k)\wedge(j\not\equiv_{\{m\}(s)}k\vee j\equiv_{\{m\}(t)}k)).$$

In order to define the positive clauses for the codes of sets we must introduce some notations. In this way we transform the clauses for realizability for sets automatically in the clauses needed to define the fix points $\mathbf{Set}(n)$, $t\overline{\varepsilon}n$, $t\,\overline{\not\varepsilon}\,n$, $t\equiv_n s$ and $t\not\equiv_n s$.

First of all, we define a function $[\,]$ which assigns a value to a set according to the table in section 3.2 as follows.

1. if σ is one of the symbols A, A', B, ϕ, ϕ', ψ, t, s, then $[\sigma]$ is a, a', $\{b\}(x)$, c, c', $\{d\}(x)$, e, f respectively;
2. if σ is N_0, N_1, N, $(\Pi x\in A)\,B$, $(\Sigma x\in A)\,B$, $A+A'$, $List(A)$ then $[\sigma]$ is $\langle 1,0\rangle$, $\langle 1,1\rangle$, $\langle 1,2\rangle$, $\langle 2,\langle a,b\rangle\rangle$, $\langle 3,\langle a,b\rangle\rangle$, $\langle 4,\langle a,a'\rangle\rangle$, $\langle 5,a\rangle$ respectively;
3. if σ is $\bot$, $\phi\wedge\phi'$, $\phi\vee\phi'$, $\phi\rightarrow\phi'$, $(\exists x\in A)\,\psi$, $(\forall x\in A)\,\psi$, $\mathsf{Id}(A,t,s)$ then $[\sigma]$ is $\langle 6,0\rangle$, $\langle 7,\langle c,c'\rangle\rangle$, $\langle 8,\langle c,c'\rangle\rangle$, $\langle 9,\langle c,c'\rangle\rangle$, $\langle 10,\langle a,d\rangle\rangle$, $\langle 11,\langle a,d\rangle\rangle$, $\langle 12,\langle a,\langle e,f\rangle\rangle\rangle$ respectively.

We denote by $[\,]^{-1}$ the inverse function of $[\,]$. Now, all clauses in the realizability interpretation of sets are defined using formulas which are obtained starting from arithmetical formulas or primitive formulas with ε or $\sim$, by using connectives, first order quantifiers or explicit instances of substitution in x. For such formulas φ we define φ^+ as follows:

1. if φ is arithmetical, then φ^+ is defined as φ itself. If φ is a primitive formulas with ε or $\sim$ we will transform $\varepsilon\,\mathcal{J}(\sigma)$ and $\sim_{\mathcal{I}(\sigma)}$ in $\overline{\varepsilon}\,[\sigma]$ and $\equiv_{[\sigma]}$ respectively, in order to obtain φ^+;
2. $(\varphi[\alpha/x])^+$ is $\varphi^+[\alpha/x]$;
3. $(\varphi\wedge\varphi')^+$ is $\varphi^+\wedge\varphi'^+$;
4. $(\varphi\vee\varphi')^+$ is $\varphi^+\vee\varphi'^+$;
5. $(\varphi\rightarrow\varphi')^+$ is $\overline{\varphi^+}\vee\varphi'^+$;
6. $(\forall u\,\varphi)^+$ is $\forall u\,\varphi^+$ for every variable u;
7. $(\exists u\,\varphi)^+$ is $\exists u\,\varphi^+$ for every variable u;

where $\overline{\varphi}$ is defined by the following clauses:

1. if φ is an arithmetical formula $\overline{\varphi}$ is $\neg\varphi$;
2. if φ is a relation between two terms through $\overline{\varepsilon}$, $\overline{\not\varepsilon}$, $\equiv$ or $\not\equiv$, then $\overline{\varphi}$ is obtained by transforming them in $\overline{\not\varepsilon}$, $\overline{\varepsilon}$, $\not\equiv$ or $\equiv$ respectively;
3. $\overline{\varphi\wedge\varphi'}$ is $\overline{\varphi}\vee\overline{\varphi'}$;
4. $\overline{\varphi\vee\varphi'}$ is $\overline{\varphi}\wedge\overline{\varphi'}$;
5. $\overline{\forall u\,\varphi}$ is $\exists u\,\overline{\varphi}$ for every variable u;
6. $\overline{\exists u\,\varphi}$ is $\forall u\,\overline{\varphi}$ for every variable u;

We can now define the positive clauses we needed. For τ equal to $\langle 1,0\rangle$, $\langle 1,1\rangle$, $\langle 1,2\rangle$, $\langle 2,\langle a,b\rangle\rangle$, $\langle 3,\langle a,b\rangle\rangle$, $\langle 4,\langle a,a'\rangle\rangle$, $\langle 5,a\rangle$, $\langle 6,0\rangle$, $\langle 7,\langle c,c'\rangle\rangle$, $\langle 8,\langle c,c'\rangle\rangle$, $\langle 9,\langle c,c'\rangle\rangle$, $\langle 10,\langle a,d\rangle\rangle$,

$\langle 11, \langle a, d\rangle\rangle$, $\langle 12, \langle a, \langle e, f\rangle\rangle\rangle$ we have the following clauses[7]:

1. $\mathbf{Set}(\tau)$ if $\mathsf{Cond}(\tau)$;
2. $n\,\bar{\varepsilon}\,\tau$ if $\mathsf{Cond}(\tau) \wedge (n\,\varepsilon\,\mathcal{J}([\tau]^{-1}))^{+}$;
3. $n\,\not\bar{\varepsilon}\,\tau$ if $\mathsf{Cond}(\tau) \wedge \overline{(n\,\varepsilon\,\mathcal{J}([\tau]^{-1}))^{+}}$;
4. $n \equiv_{\tau} m$ if $\mathsf{Cond}(\tau) \wedge (n \sim_{\mathcal{I}([\tau]^{-1})} m)^{+}$;
5. $n \not\equiv_{\tau} m$ if $\mathsf{Cond}(\tau) \wedge \overline{(n \sim_{\mathcal{I}([\tau]^{-1})} m)^{+}}$;

where $\mathsf{Cond}(\tau)$ is

1. $\top$ if τ has first component 1 or 6;
2. $\mathbf{Fam}(b, a)$ if τ has first component 2 or 3;
3. $\mathbf{Set}(a) \wedge \mathbf{Set}(a')$ if τ has first component 4;
4. $\mathbf{Set}(a)$ if τ has first component 5;
5. $\mathbf{Set}(c) \wedge \mathbf{Set}(c') \wedge \pi_1(c) > 5 \wedge \pi_1(c') > 5$ if τ has first component 7, 8 or 9;
6. $\mathbf{Fam}(d, a) \wedge \forall x\,(x\,\not\bar{\varepsilon}\,a \vee \pi_1(\{d\}(x)) > 5)$ if τ has first component 10, 11;
7. $\mathbf{Set}(a) \wedge e\,\bar{\varepsilon}\,a \wedge f\,\bar{\varepsilon}\,a$ if τ has first component 12.

By sake of example we present here the clauses for codes of Π-sets.

$\mathbf{Set}(\langle 2, \langle a, b\rangle\rangle)$ if $\mathbf{Fam}(b, a)$;
$n\,\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle$ if

$$\mathbf{Fam}(b, a) \wedge \forall x\,(x\,\not\bar{\varepsilon}\,a \vee \{n\}(x)\,\bar{\varepsilon}\,\{b\}(x)) \wedge \forall x \forall y\,(x \not\equiv_a y \vee \{n\}(x) \equiv_{\{b\}(x)} \{n\}(y));$$

$n\,\not\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle$ if

$$\mathbf{Fam}(b, a) \wedge (\exists x\,(x\,\bar{\varepsilon}\,a \wedge \{n\}(x)\,\not\bar{\varepsilon}\,\{b\}(x)) \vee \exists x \exists y\,(x \equiv_a y \wedge \{n\}(x) \not\equiv_{\{b\}(x)} \{n\}(y)));$$

$n \equiv_{\langle 2, \langle a, b\rangle\rangle} m$ if

$$\mathbf{Fam}(b, a) \wedge n\,\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle \wedge m\,\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle \wedge \forall x\,(x\,\not\bar{\varepsilon}\,a \vee \{n\}(x) \equiv_{\{b\}(x)} \{m\}(x));$$

$n \not\equiv_{\langle 2, \langle a, b\rangle\rangle} m$ if

$$\mathbf{Fam}(b, a) \wedge (n\,\not\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle \vee m\,\not\bar{\varepsilon}\,\langle 2, \langle a, b\rangle\rangle \vee \exists x\,(x\,\bar{\varepsilon}\,a \wedge \{n\}(x) \not\equiv_{\{b\}(x)} \{m\}(x))$$

The formulas $\mathbf{Set}(n)$, $t\bar{\varepsilon}n$, $t\,\not\bar{\varepsilon}\,n$, $t \equiv_n s$ and $t \not\equiv_n s$ are components of a predicate $P_\theta(n)$ defined in $\widehat{ID_1}$ as a fix point of an operator $\theta(n, X)$ defined by glueing together the clauses expressing the code of each $\mathbf{mTT}^s$-set-constructor with its interpretation in $\widehat{ID_1}$.

The interpretation of collections

Here we extend the realizability relation, membership and equality in definition 5 in order to interpret collections, propositions and the decoding operators.

[7] By $n\varepsilon\mathcal{J}([\tau]^{-1})$ and $n \sim_{\mathcal{I}([\tau]^{-1})} m$ we mean the right-hand side of the respective clause in the realizability interpretation of sets, taking into account that for small propositions membership coincides with the realizability relation.

▶ **Definition 6.** $n \Vdash \phi$ between natural numbers and $\mathbf{mTT}^s$-propositions and formulas $n\varepsilon\mathcal{J}(D)$ and $n \sim_{\mathcal{I}(D)} m$ for collections D are defined by including all clauses in definition 5 plus the following:

1. $n \Vdash \tau(p)$ and $n\,\varepsilon\,\mathcal{J}(\tau(p))$ are both given by $n\,\overline{\varepsilon}\,\mathcal{I}(p)$
$n \sim_{\mathcal{I}(\tau(p))} m$ is $n\,\varepsilon\,\mathcal{J}(\tau(p)) \wedge m\,\varepsilon\,\mathcal{J}(\tau(p))$;

2. The realizability relation $n \Vdash \eta$ for propositions is completely analogous to the realizability relation for small propositions and the interpretation of propositions is given by the class of realizers equipped with the trivial equivalence relation;

3. Σ-collections are interpreted exactly in the same way as Σ-sets;

4. $n\varepsilon\mathcal{J}(\mathsf{Set})$ is $\mathbf{Set}(n) \wedge \forall t\,(t\,\overline{\varepsilon}\,n \leftrightarrow \neg t \not\overline{\varepsilon}\, n) \wedge \forall t \forall s\,(t \equiv_n s \leftrightarrow \neg t \not\equiv_n s)$. This is because $\not\overline{\varepsilon}$ and $\not\equiv$, which are defined by fix point, don't behave necessarily as negations of $\overline{\varepsilon}$ and $\equiv$ and hence we need to add $\forall t\,(t\,\overline{\varepsilon}\,n \leftrightarrow \neg t \not\overline{\varepsilon}\, n)$ and $\forall t \forall s\,(t \equiv_n s \leftrightarrow \neg t \not\equiv_n s)$;
The interpretation of $n \sim_{\mathcal{I}(\mathsf{Set})} m$ is
$$n\,\varepsilon\,\mathcal{J}(\mathsf{Set}) \wedge m\,\varepsilon\,\mathcal{J}(\mathsf{Set}) \wedge \forall t\,(t\,\overline{\varepsilon}\,n \leftrightarrow t\,\overline{\varepsilon}\,m) \wedge \forall t \forall s\,(t \equiv_n s \leftrightarrow t \equiv_m s);$$

5. $n\,\varepsilon\,\mathcal{J}(\mathsf{props_s})$ is $n\,\varepsilon\,\mathcal{J}(\mathsf{Set}) \wedge \pi_1(n) > 5 \wedge \forall t \forall s\,(t\,\overline{\varepsilon}\,n \wedge s\,\overline{\varepsilon}\,n \leftrightarrow t \equiv_n s)$ (recall that small propositions are encoded with $\pi_1(n) > 5$ and enjoy the proof-irrelevance);
The interpretation of $n \sim_{\mathcal{I}(\mathsf{props_s})} m$ is $n\,\varepsilon\,\mathcal{J}(\mathsf{props_s}) \wedge m\,\varepsilon\,\mathcal{J}(\mathsf{props_s}) \wedge \forall t\,(t\,\overline{\varepsilon}\,n \leftrightarrow t\,\overline{\varepsilon}\,m)$;

6. $n\,\varepsilon\,\mathcal{J}(A \to \mathsf{props_s})$ is $\forall t \forall s\,(t \sim_{\mathcal{I}(A)} s \to \{n\}(t) \sim_{\mathcal{I}(\mathsf{props_s})} \{n\}(s))$
and $n \sim_{\mathcal{I}(A \to \mathsf{props_s})} m$ is
$n\,\varepsilon\,\mathcal{J}(A \to \mathsf{props_s}) \wedge m\,\varepsilon\,\mathcal{J}(A \to \mathsf{props_s}) \wedge \forall t\,(t\,\varepsilon\,\mathcal{J}(A) \to \{n\}(t) \sim_{\mathcal{I}(\mathsf{props_s})} \{m\}(t))$

▶ Remark. Notice that the properties of sets and small propositions stated after definition 5 hold also for collections and propositions respectively.

The interpretation of judgements

We now need to say how judgements are interpreted in our realizability model. First of all, if $\mathcal{A} = (A, \simeq_{\mathcal{A}})$ and $\mathcal{B} = (B, \simeq_{\mathcal{B}})$ are definable classes of $\widehat{ID_1}$ equipped with a definable equivalence relation, then we denote with $\mathcal{A} \doteq \mathcal{B}$ the formula $\forall t \forall s\,(t \simeq_{\mathcal{A}} s \leftrightarrow t \simeq_{\mathcal{B}} s)$.

The judgements of $\mathbf{mTT}^s$ are interpreted as follows:

1. if $type \in \{set, col, props_s, prop\}$, the interpretation of $A\ type$ is $\mathcal{I}(A) \doteq \mathcal{I}(A)$;
2. if $type \in \{set, col, props_s, prop\}$, the interpretation of $A = B\ type$ is $\mathcal{I}(A) \doteq \mathcal{I}(B)$;
3. the judgement $t \in A$ is interpreted as $\mathcal{I}(t)$;
4. the judgement $t = s \in A$ is interpreted as $\mathcal{I}(t) \sim_{\mathcal{I}(A)} \mathcal{I}(s)$.

5. if $type \in \{set, col, props_s, prop\}$, the interpretation of $A\ type\ [x_1 \in A_1, ..., x_{\mathsf{n}} \in A_{\mathsf{n}}]$ is
$$\forall x_1 \forall y_1 ... \forall x_{\mathsf{n}} \forall y_{\mathsf{n}}\,(x_1 \sim_{\mathcal{I}(A_1)} y_1 \wedge ... \wedge x_{\mathsf{n}} \sim_{\mathcal{I}(A_{\mathsf{n}})} y_{\mathsf{n}} \to \mathcal{I}(A) \doteq \mathcal{I}(A)\,[y_1/x_1, ..., y_{\mathsf{n}}/x_{\mathsf{n}}])$$ [8]

[8] Note that this definition and the following exploit the fact that $\mathbf{mTT}^s$-variables are interpreted as themselves thought of as $\widehat{ID_1}$-variables.

6. if *type* $\in \{set, col, props_s, prop\}$, the interpretation of $A = B\ type\ [x_1 \in A_1, ..., x_{\mathsf{n}} \in A_{\mathsf{n}}]$ is

$$\forall x_1 ... \forall x_{\mathsf{n}}\, (x_1\, \varepsilon\, \mathcal{J}(A_1)\ \wedge ... \wedge\ x_{\mathsf{n}}\, \varepsilon\, \mathcal{J}(A_{\mathsf{n}})\ \rightarrow\ \mathcal{I}(A) \doteq \mathcal{I}(B))$$

7. the judgement $t \in A[x_1 \in A_1, ..., x_{\mathsf{n}} \in A_{\mathsf{n}}]$ is interpreted as

$$\forall x_1 \forall y_1 ... \forall x_{\mathsf{n}} \forall y_{\mathsf{n}}\, (x_1 \sim_{\mathcal{I}(A_1)} y_1\ \wedge ... \wedge\ x_{\mathsf{n}} \sim_{\mathcal{I}(A_{\mathsf{n}})} y_{\mathsf{n}}\ \rightarrow\ \mathcal{I}(t) \sim_{\mathcal{I}(A)} \mathcal{I}(t)\, [y_1/x_1, .., y_{\mathsf{n}}/x_{\mathsf{n}}])$$

8. the judgement $t = s \in A[x_1 \in A_1, ..., x_{\mathsf{n}} \in A_{\mathsf{n}}]$ is interpreted as

$$\forall x_1 ... \forall x_{\mathsf{n}}\, (x_1\, \varepsilon\, \mathcal{J}(A_1)\ \wedge ... \wedge\ x_{\mathsf{n}}\, \varepsilon\, \mathcal{J}(A_{\mathsf{n}})\ \rightarrow\ \mathcal{I}(t) \sim_{\mathcal{I}(A)} \mathcal{I}(s))$$

3.3 The validity theorem

A judgement J in the language of $\mathbf{mTT}^s$ is validated by the realizability model ($\mathcal{R} \vDash J$) if $\widehat{ID_1} \vdash \mathcal{I}(J)$, where $\mathcal{I}(J)$ is the interpretation of J according to the previous section. We say that a proposition ϕ is validated by the model ($\mathcal{R} \vDash \phi$), if its interpretation can be proven to be inhabited, which means that

$\widehat{ID_1} \vdash \exists r(r \varepsilon \mathcal{J}(\phi))$ which is equivalent to $\widehat{ID_1} \vdash \exists r(r \Vdash \phi)$.

In order to prove how substitution is interpreted in a easy way, it is convenient to modify the presentation of $\mathbf{mTT}^s$-rules, into an equivalent system (still denoted by $\mathbf{mTT}^s$), where we supply the information that the members in a type equality judgement are types, and members of term equality judgements are typed terms as follows with the warning of avoiding repetitions of same judgements: for *type* $\in \{set, col, props_s, prop\}$

any rule $\dfrac{J_1 ... J_n}{A = B\ type\ [\Gamma]}$ is changed to $\dfrac{J_1 ... J_n,\ A\ type\ [\Gamma]\ ,\ B\ type\ [\Gamma]}{A = B\ type\ [\Gamma]}$

any rule $\dfrac{J_1 ... J_n}{b \in B\ [\Gamma]}$ is changed to $\dfrac{J_1 ... J_n,\ B\ type\ [\Gamma]}{b \in B\ [\Gamma]}$

any rule $\dfrac{J_1 ... J_n}{a = b \in A\ [\Gamma]}$ is changed to $\dfrac{J_1 ... J_n,\ a \in A\ type\ [\Gamma]\ ,\ b \in A\ type\ [\Gamma]}{a = b \in A\ type\ [\Gamma]}$

the substitution rule subT) and sub) in [9] are changed to

$$\text{subT}_m)\ \frac{\begin{array}{l} C(x_1, \ldots, x_n)\ type\ [\, x_1 \in A_1,\ \ldots,\ x_n \in A_n(x_1, \ldots, x_{n-1})\,] \\ a_1 \in A_1,\ \ldots\ , a_n \in A_n(a_1, \ldots, a_{n-1}) \qquad b_1 \in A_1,\ \ldots\ , b_n \in A_n(b_1, \ldots, b_{n-1}) \\ a_1 = b_1 \in A_1\ \ldots\ a_n = b_n \in A_n(a_1, \ldots, a_{n-1}) \end{array}}{C(a_1, \ldots, a_n) = C(b_1, \ldots, b_n)\ type}$$

$$\text{sub}_m)\ \frac{\begin{array}{l} c(x_1, \ldots, x_n) \in C(x_1, \ldots, x_n)\ \ [\, x_1 \in A_1,\ \ldots,\ x_n \in A_n(x_1, \ldots, x_{n-1})\,] \\ C(x_1, \ldots, x_n)\ type\ [\, x_1 \in A_1,\ \ldots,\ x_n \in A_n(x_1, \ldots, x_{n-1})\,] \\ a_1 \in A_1,\ \ldots\ , a_n \in A_n(a_1, \ldots, a_{n-1}) \qquad b_1 \in A_1,\ \ldots\ , b_n \in A_n(b_1, \ldots, b_{n-1}) \\ a_1 = b_1 \in A_1\ \ldots\ a_n = b_n \in A_n(a_1, \ldots, a_{n-1}) \end{array}}{c(a_1, \ldots, a_n) = c(b_1, \ldots, b_n) \in C(a_1, \ldots, a_n)}$$

the formation rules F-Σ), F-∃) and F-∀) are changed to

$$\text{F-}\Sigma)\ \frac{\begin{array}{l} B\ col \\ C(x)\ \ col\ [x \in B] \end{array}}{\Sigma_{x \in B} C(x)\ col} \qquad \text{F-}\exists)\ \frac{\begin{array}{l} B\ col \\ C(x)\ prop\ \ [x \in B] \end{array}}{\exists_{x \in B} C(x)\ prop} \qquad \text{F-}\forall)\ \frac{\begin{array}{l} B\ col \\ C(x)\ prop\ [x \in B] \end{array}}{\forall_{x \in B} C(x)\ prop}$$

the elimination rules E-Π) and E-∀) are changed to

$$\text{E-}\Pi_m)\ \frac{\begin{array}{c}C(x)\ set\,[x\in B]\quad C(b)\ set\\ b\in B\quad f\in \Pi_{x\in B}C(x)\end{array}}{\mathsf{Ap}(f,b)\in C(b)}\qquad \text{E-}\forall_m)\ \frac{\begin{array}{c}C(x)\ prop\,[x\in B]\quad C(b)\ prop\\ b\in B\quad f\in \forall_{x\in B}C(x)\end{array}}{\mathsf{Ap}_\forall(f,b)\in C(b)}$$

Note that each $\mathbf{mTT}^s$-type is a collection and therefore in deriving a typed term $b\in B$ under a context the addition of the information that the type B is a collection in the premise is certainly valid.

▶ **Theorem 7** (Validity theorem). *For every judgement J in the language of $\mathbf{mTT}^s$, if J can be proven in $\mathbf{mTT}^s$ ($\mathbf{mTT}^s \vdash J$), then J is validated by the model ($\mathcal{R}\vDash J$).*

Proof. In order to prove the validity theorem it is necessary to prove by induction on the height of the proof tree in $\mathbf{mTT}^s$ these three facts at the same time:

1. for every judgement J in the language of $\mathbf{mTT}^s$, if $\mathbf{mTT}^s\vdash J$ then $\mathcal{R}\vDash J$;
2. (substitution) If $\mathbf{mTT}^s\vdash C\,type\,[x_1\in A_1,...,x_\mathsf{n}\in A_\mathsf{n}]$ for $type\in\{set, col, props_s, prop\}$ for all

$$\mathbf{mTT}^s\vdash a_1\in A_1[y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],...,$$
$$\mathbf{mTT}^s\vdash a_\mathsf{n}\in A_\mathsf{n}[a_1/x_1,...,a_{\mathsf{n}-1}/x_{\mathsf{n}-1}][y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],$$

if $\mathcal{R}\vDash a_1\in A_1[y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}]$,...,

$$\mathcal{R}\vDash a_\mathsf{n}\in A_\mathsf{n}[a_1/x_1,...,a_{\mathsf{n}-1}/x_{\mathsf{n}-1}][y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],$$

then

$$\widehat{ID_1}\vdash\forall y_1...\forall y_\mathsf{m}\,(y_1\,\varepsilon\,\mathcal{J}(B_1)\wedge...\wedge y_\mathsf{m}\,\varepsilon\,\mathcal{J}(B_\mathsf{m})\rightarrow$$
$$\mathcal{I}(C)\,[\mathcal{I}(a_1)/x_1,...,\mathcal{I}(a_\mathsf{n})/x_\mathsf{n}]\doteq\mathcal{I}(C\,[a_1/x_1,...,a_\mathsf{n}/x_\mathsf{n}]))$$

and if $\mathbf{mTT}^s\vdash c\in C[x_1\in A_1,...,x_\mathsf{n}\in A_\mathsf{n}]$ for all

$$\mathbf{mTT}^s\vdash a_1\in A_1[y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],...,$$
$$\mathbf{mTT}^s\vdash a_\mathsf{n}\in A_\mathsf{n}[a_1/x_1,...,a_{\mathsf{n}-1}/x_{\mathsf{n}-1}][y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],$$

if $\mathcal{R}\vDash a_1\in A_1[y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}]$,...,

$$\mathcal{R}\vDash a_\mathsf{n}\in A_\mathsf{n}[a_1/x_1,...,a_{\mathsf{n}-1}/x_{\mathsf{n}-1}][y_1\in B_1,...,y_\mathsf{m}\in B_\mathsf{m}],$$

then

$$\widehat{ID_1}\vdash\forall y_1...\forall y_\mathsf{m}\,(y_1\,\varepsilon\mathcal{J}(B_1)\wedge...\wedge y_\mathsf{m}\,\varepsilon\,\mathcal{J}(B_\mathsf{m})\rightarrow$$
$$\mathcal{I}(c)\,[\mathcal{I}(a_1)/x_1,...,\mathcal{I}(a_\mathsf{n})/x_\mathsf{n}]\sim_{\mathcal{I}(C\,[a_1/x_1,...,a_\mathsf{n}/x_\mathsf{n}])}\mathcal{I}(c\,[a_1/x_1,...,a_\mathsf{n}/x_\mathsf{n}])).$$

3. (coding) If $\mathbf{mTT}^s\vdash B\,set\,[x_1\in A_1,...,x_\mathsf{n}\in A_\mathsf{n}]$, then

$$\widehat{ID_1}\vdash\forall x_1...\forall x_\mathsf{n}\,(x_1\,\varepsilon\,\mathcal{J}(A_1)\wedge...\wedge x_\mathsf{n}\,\varepsilon\,\mathcal{J}(A_\mathsf{n})\rightarrow\mathbf{Set}(\mathcal{I}(\hat{B}))\wedge$$
$$\forall t\,(t\,\varepsilon\,\mathcal{J}(B)\leftrightarrow t\,\overline{\varepsilon}\,\mathcal{I}(\hat{B}))\wedge\forall t\,(\neg t\,\varepsilon\,\mathcal{J}(B)\leftrightarrow t\,\overline{\not\varepsilon}\,\mathcal{I}(\hat{B}))\wedge$$
$$\forall t\forall s\,(t\sim_{\mathcal{I}(B)}s\leftrightarrow t\equiv_{\mathcal{I}(\hat{B})}s)\wedge\forall t\forall s\,(\neg t\sim_{\mathcal{I}(B)}s\leftrightarrow t\not\equiv_{\mathcal{I}(\hat{B})}s)).$$

Let us show only some cases, as the techniques will be similar in the other cases. In particular we will consider validity and substitution for the rule of introduction and validity for the rule of conversion for Π-sets. The technique is similar in the other cases.

1. Suppose that we derived in $\mathbf{mTT}^s$ the judgement $\lambda y.c \in (\Pi y \in B)\, C\, [x \in A]$ by introduction rule, after having derived $c \in C\, [x \in A, y \in B]$ and $(\Pi y \in B)\, C\, set\, [x \in A]$. By inductive hypothesis on validity we can suppose that $\mathcal{R} \vDash c \in C\, [x \in A, y \in B]$. This means that in $\widehat{ID_1}$
$$\forall x \forall x' \forall y \forall y'\, (x \sim_{\mathcal{I}(A)} x' \wedge y \sim_{\mathcal{I}(B)} y' \rightarrow \mathcal{I}(c) \sim_{\mathcal{I}(C)} \mathcal{I}(c)\, [x'/x, y'/y]\,)$$
which is equivalent to
$$\forall x \forall x'\, (x \sim_{\mathcal{I}(A)} x' \rightarrow \forall y \forall y'\, (y \sim_{\mathcal{I}(B)} y' \rightarrow \{\Lambda x.\Lambda y.\mathcal{I}(c)\}(x, y) \sim_{\mathcal{I}(C)} \{\Lambda x.\Lambda y.\mathcal{I}(c)\}(x', y'))$$
Using this fact together with the fact that, by inductive hypothesis on validity (as the judgement $B\, set\, [x \in A]$ is derived with a shorter derivation), in $\widehat{ID_1}$ we have that $\forall x \forall x'\, (x \sim_{\mathcal{I}(A)} x' \rightarrow \mathcal{I}(B) \doteq \mathcal{I}(B)[x'/x])$, we obtain that
$$\widehat{ID_1} \vdash \forall x \forall x'\, (x \sim_{\mathcal{I}(A)} x' \rightarrow \mathcal{I}(\lambda y.c) \sim_{\mathcal{I}((\Pi y \in B) C)} \mathcal{I}(\lambda y.c)\, [x'/x]).$$
2. We want to prove the substitution property for a judgement $\lambda y.c \in (\Pi y \in B)\, C\, [x \in A]$ derived in $\mathbf{mTT}$ by introduction after having derived $c \in C\, [x \in A, y \in B]$ and $(\Pi y \in B)\, C\, set\, [x \in A]$ with respect to a term a for which $\mathbf{mTT} \vdash a \in A$ and $\mathcal{R} \vDash a \in A$. First of all notice that by the structure of derivations in $\mathbf{mTT}$, $B\, set\, [x \in A]$ and (so also) $B[a/x]\, set$ can be derived in $\mathbf{mTT}$. From this it follows that $\mathbf{mTT} \vdash a \in A[y \in B[a/x]]$ and $\mathbf{mTT} \vdash y \in B[a/x]\, [y \in B[a/x]]$. These judgements are also validated by $\mathcal{R}$ by definition of the interpretation and because $\mathcal{R} \vDash a \in A$. By inductive hypothesis on substitution applied to $c \in C\, [x \in A, y \in B]$ with respect to $a \in A[y \in B[a/x]]$ and $y \in B[a/x]\, [y \in B[a/x]]$ we obtain that in $\widehat{ID_1}$ it holds that
$\forall y\, (y \varepsilon \mathcal{J}(B[a/x]) \rightarrow \mathcal{I}(c)[\mathcal{I}(a)/x] \sim_{\mathcal{I}(C[a/x])} \mathcal{I}(c[a/x]))$ and this entails that
$$\widehat{ID_1} \vdash \forall y\, (y\, \varepsilon\, \mathcal{J}(B[a/x]) \rightarrow \{\mathcal{I}(\lambda y.c)[\mathcal{I}(a)/x]\}(y) \sim_{\mathcal{I}(C[a/x])} \{\mathcal{I}(\lambda y.c[a/x])\}(y)).$$
In order to conclude it is easy to prove, by using suitable inductive hypotheses that
$$\widehat{ID_1} \vdash \mathcal{I}(\lambda y.c)[\mathcal{I}(a)/x]\, \varepsilon\, \mathcal{I}((\Pi y \in B)\, C[a/x]) \wedge \mathcal{I}(\lambda y.c[a/x])\, \varepsilon\, \mathcal{I}((\Pi y \in B)\, C[a/x]).$$
3. If the judgement $\mathsf{Ap}(\lambda y.b, a) = b[a/x] \in B[a/x]$ is derived in $\mathbf{mTT}^s$ by conversion after having derived $b \in B\, [x \in A]$, $a \in A$, $b[a/x] \in B[a/x]$ and $\mathsf{Ap}(\lambda y.b, a) \in B[a/x]$, in order to show that it is validated by $\mathcal{R}$, we can suppose by inductive hypothesis on validity that $\mathcal{R} \vDash b \in B\, [x \in A]$ and $\mathcal{R} \vDash a \in A$.
By inductive hypothesis on substitution applied to $b \in B\, [x \in A]$ with respect to $a \in A$ we obtain that $\widehat{ID_1} \vdash \mathcal{I}(b)\, [\mathcal{I}(a)/x] \sim_{\mathcal{I}(B\, [a/x])} \mathcal{I}(b\, [a/x])$. But by the definition of the interpretation of $\mathsf{Ap}(\lambda y.b, a)$ from this we obtain that
$$\widehat{ID_1} \vdash \mathcal{I}(\mathsf{Ap}(\lambda y.b, a)) \sim_{\mathcal{I}(B\, [a/x])} \mathcal{I}(b\, [a/x])$$
which exactly means that $\mathcal{R} \vDash \mathsf{Ap}(\lambda y.b, a) = b\, [a/x] \in B\, [a/x]$.

For rules about prop_s, the definitions of $\mathcal{I}(\mathsf{prop}_s)$ and $\mathcal{I}(\tau(p))$ were given in such a way that validity and substitution can be checked easily, sometimes (e. g. in the case of quantifiers $\widehat{(\exists x \in A)\, p}$ and $\widehat{(\forall x \in A)\, p}$) using the inductive hypothesis (coding), which guarantees that if you start from A which is proven to be a set in $\mathbf{mTT}^s$ and you perform the coding in the syntax $\widehat{A}$, then $\widehat{A}$ is a well defined code for a set and it is exactly the internal version of $\mathcal{I}(\mathsf{A})$.
◀

Consequences of the validity theorem

We discuss here about the validity in our realizability model for **mTT** of some principles, namely Extensionality Equality of Functions, Axiom of Choice and formal Church Thesis.

1. **Extensionality Equality of Functions** can be formulated as a proposition in **mTT** as follows:

$$(\mathbf{extFun}) \quad (\forall f \in (\Pi x \in A)\, B)\, (\forall g \in (\Pi x \in A)\, B) \\ ((\forall x \in A)\, \mathsf{Id}(B, \mathsf{Ap}(f,x), \mathsf{Ap}(g,x)) \to \mathsf{Id}((\Pi x \in A)\, B, f, g))$$

Since the judgements $f = g \in (\Pi x \in A)\, B$ and $\mathsf{Ap}(f,x) = \mathsf{Ap}(g,x) \in B\, [x \in A]$ have the same interpretation, **extFun** can be realized by the term $\Lambda f.\Lambda g.\Lambda r.0$, i. e. our model realises **extFun**.

2. The **Axiom of Choice** $\mathbf{AC}_{A,B}$ is represented in **mTT** by the following proposition:

$$(\mathbf{AC}_{A,B})\ (\forall x \in A)\, (\exists y \in B)\, \rho(x,y) \to (\exists f \in (\Pi x \in A)\, B)\, (\forall x \in A)\, \rho(x, \mathsf{Ap}(f,x))$$

Unfortunately if $\rho(x,y)$ is a proposition for which $\mathcal{R} \vDash \rho(x,y)\, prop\, [x \in A,\, y \in B]$, a realizer r for $(\forall x \in A)\, (\exists y \in B)\, \rho(x,y)$ cannot be turned into a recursive function from $\mathcal{J}(A)$ to $\mathcal{J}(B)$ respecting equivalence relations $\sim_{\mathcal{I}(A)}$ and $\sim_{\mathcal{I}(B)}$, as the interpretation of propositions is proof-irrelevant and we can have different elements a and a' of $\mathcal{J}(A)$ which are equivalent in $\mathcal{I}(A)$ for which $\pi_1(\{r\}(a))$ and $\pi_1(\{r\}(a'))$ are not equivalent in $\mathcal{I}(B)$. This problem can be avoided if A is a numerical set and in particular in the case of the set N. In this case the natural number $\Lambda r.\langle \Lambda n.\pi_1(\{r\}(n)), \Lambda n.\pi_2(\{r\}(n))\rangle$ is a realizer for the axiom of choice $\mathbf{AC}_{N,B}$. So $\mathcal{R} \Vdash \mathbf{AC}_{N,B}$ for every B.
Moreover also the axiom of unique choice $\mathbf{AC}_!$ given by

$$(\mathbf{AC}_!)\ (\forall x \in A)\, (\exists! y \in B)\, \rho(x,y) \to (\exists f \in (\Pi x \in A)\, B)\, (\forall x \in A)\, \rho(x, \mathsf{Ap}(f,x))$$

is validated by the model $\mathcal{R}$.[9] In fact if ρ is a proposition for which we have that $\mathcal{R} \vDash \rho(x,y)\, prop\, [x \in A,\, y \in B]$, then in particular

$$\widehat{ID_1} \vdash \forall x \forall x' \forall y \forall t\, (x \sim_{\mathcal{I}(A)} x' \,\wedge\, y\, \varepsilon\, \mathcal{J}(B) \,\wedge\, t \Vdash \rho(x,y) \to t \Vdash \rho(x', y)).$$

This implies that we can easily choose a realizer for the axiom of unique choice.

3. If φ is a formula of first-order arithmetic HA, then we can define a proposition $\overline{\varphi}$ in **mTT**, according to the following conditions:

$\overline{\bot}$ is $\bot$	$\overline{\varphi \wedge \varphi'}$ is $\overline{\varphi} \wedge \overline{\varphi'}$	$\overline{\varphi \to \varphi'}$ is $\overline{\varphi} \to \overline{\varphi'}$	$\overline{\exists x\, \varphi}$ is $(\exists x \in N)\, \overline{\varphi}$
$\overline{t = s}$ is $\mathsf{Id}(N, \overline{t}, \overline{s})$	$\overline{\varphi \vee \varphi'}$ is $\overline{\varphi} \vee \overline{\varphi'}$		$\overline{\forall x\, \varphi}$ is $(\forall x \in N)\, \overline{\varphi}$

where $\overline{t}$ and $\overline{s}$ are the translations of terms of HA in **mTT** (in particular primitive recursive functions of HA are translated via El_N, $succ$ and 0 are translated in the obvious corresponding ones and variables are interpreted as themselves[10]). The language of HA can also be naturally interpreted in $\widehat{ID_1}$ by using the fact that each primitive recursive function can be encoded by a numeral. If t is a term of HA we will still write t for its translation in $\widehat{ID_1}$. The following lemma is an immediate consequence of the definition of our realizability interpretation where $\Vdash_k$ denotes Kleene realizability in HA (see [20]):

[9] $(\exists! x \in A)P(x)$ is defined as $(\exists x \in A)P(x) \wedge (\forall x \in A)(\forall x' \in A)(P(x) \wedge P(x') \to \mathsf{Id}(A, x, x'))$.
[10] Here we suppose that variables of HA coincides with variables of the untyped syntax of $\mathbf{mTT}^s$.

▶ **Lemma 8.** *If t is a term of* HA *and φ is a formula of* HA*, then*
a. $\widehat{ID_1} \vdash \mathcal{I}(\overline{t}) = t$
b. $\widehat{ID_1} \vdash n \Vdash_k \varphi \leftrightarrow n \Vdash \overline{\varphi}$.

The **formal Church Thesis CT** can be expressed in **mTT** as the following proposition

$$(\mathbf{CT})\, (\forall x \in N)\, (\exists y \in N)\, \rho(x,y) \rightarrow (\exists e \in N)\, (\forall x \in N)\, (\exists u \in N)\, (\overline{T(e,x,u)} \wedge \rho(x, \overline{U(u)}))$$

where T and U are the Kleene predicate and the primitive recursive function representing Kleene application in HA. Note that the validity of **CT** can be obtained by glueing $\mathbf{AC}_{N,N}$ together with the following restricted form of Church Thesis for type-theoretic functions:

$$(\mathbf{CT}_\lambda)\, (\forall f \in (\Pi x \in N)N)\, (\exists e \in N)\, (\forall x \in N)\, (\exists u \in N)\, (\overline{T(e,x,u)} \wedge \mathsf{Id}(N, \mathsf{Ap}(f,x), \overline{U(u)}))$$

We know by general results on Kleene realizability that there exists a numeral $\mathbf{r}$ for which $\mathsf{HA} \vdash \exists u\, T(f,x,u) \rightarrow (\{\mathbf{r}\}(f,x) \Vdash \exists u\, T(f,x,u))$. Using this remark, the fact that $\{f\}(x)\downarrow$ is equivalent to $\exists u\, T(f,x,u)$ in $\widehat{ID_1}$, the proof irrelevance and lemma 8 we can show that $\mathbf{CT}_\lambda$ can be realized by

$$\Lambda f. \langle f, \Lambda x. \langle \{\mathbf{p}_1\}(\{\mathbf{r}\}(f,x)), \langle \{\mathbf{p}_2\}(\{\mathbf{r}\}(f,x)), 0 \rangle \rangle \rangle.$$

In fact every function from N to N is interpreted in the model as a code for a total recursive function and we can send this code to itself in order to realize Church Thesis. *Proof irrelevance allows to ignore the problem that different codes can give rise to extensionally equal functions, which is crucial to prove validity of* **CT**.
We can conclude this section by stating the following consistency results:

▶ **Theorem 9.** **mTT** *is consistent with* **CT**.
▶ **Corollary 10.** **emTT** *is consistent with* **CT**.

Proof. According to the interpretation of **emTT** in **mTT** in [9], the interpretation of **CT** turns now to be equivalent to **CT** itself. Therefore a model showing consistency of **mTT** with **CT** can be extended to a model of **emTT** with **CT**. ◀

4 Conclusions

As explained in the introduction, the semantics built here seems to be the best approximation of Kleene realizability for the extensional level **emTT** of the Minimalist Foundation, since **emTT** validates Extensionality Equality of Functions and it is constructively incompatible with the Axiom of Choice on generic sets (see [9]), which is instead valid in Beeson's model. In our semantics instances of the axiom of choice are still valid only on numerical sets, which include the interpretation of basic intensional types as the set of natural numbers.

On the contrary, for the intensional level **mTT** of the Minimalist Foundation we hope to build a more intensional realizability semantics à la Kleene where we validate not only **CT** but also the Axiom of Choice **AC** on generic types. Recalling from [9] that our **mTT** can be naturally interpreted in Martin-Löf's type theory with one universe, such an intensional Kleene realizability for **mTT** could be obtained by modelling intensional Martin-Löf's type theory with one universe (with explicit substitutions in place of the usual substitution term equality rules) together with **CT**. However, as far as we know, the consistency of intensional

Martin-Löf's type theory with **CT** is still an open problem.

Acknowledgements. We acknowledge many useful fruitful discussions with Takako Nemoto on realizability models for Martin-Löf's type theory during her visits to our department. We also thank Laura Crosilla, Giovanni Sambin and Thomas Streicher for other interesting fruitful discussions on topics of this paper. We are grateful to Ferruccio Guidi for its constant help with typesetting.

References

1 G. Barthes, V. Capretta, and O. Pons. Setoids in type theory. *J. Funct. Programming*, 13(2):261–293, 2003. Special issue on "Logical frameworks and metalanguages".

2 M. Beeson. *Foundations of Constructive Mathematics.* Springer-Verlag, Berlin, 1985.

3 E. Bishop. *Foundations of Constructive Analysis.* McGraw-Hill Book Co., 1967.

4 T. Coquand. Metamathematical investigation of a calculus of constructions. In P. Odifreddi, editor, *Logic in Computer Science*, pages 91–122. Academic Press, 1990.

5 S. Feferman. Iterated inductive fixed-point theories: application to Hancock's conjecture. In *Patras Logic Symposion*, pages 171–196. North Holland, 1982.

6 M. Hofmann. *Extensional Constructs in Intensional Type Theory.* Distinguished Dissertations. Springer, 1997.

7 J. M. E. Hyland. The effective topos. In *The L.E.J. Brouwer Centenary Symposium (Noordwijkerhout, 1981)*, volume 110 of *Stud. Logic Foundations Math.*, pages 165–216. North-Holland, Amsterdam-New York,, 1982.

8 J. M. E. Hyland, P. T. Johnstone, and A. M. Pitts. Tripos theory. *Bull. Austral. Math. Soc.*, 88:205–232, 1980.

9 M. E. Maietti. A minimalist two-level foundation for constructive mathematics. *Annals of Pure and Applied Logic*, 160(3):319–354, 2009.

10 M. E. Maietti and G. Rosolini. Elementary quotient completion. *Theory and Applications of Categories*, 27(17):445–463, 2013.

11 M. E. Maietti and G. Rosolini. Quotient completion for the foundation of constructive mathematics. *Logica Universalis*, 7(3):371–402, 2013.

12 M. E. Maietti and G. Rosolini. Unifying exact completions. *Applied Categorical Structures*, DOI 10.1007/s10485-013-9360-5, 2013.

13 M. E. Maietti and G. Sambin. Toward a minimalist foundation for constructive mathematics. In L. Crosilla and P. Schuster, editor, *From Sets and Types to Topology and Analysis: Practicable Foundations for Constructive Mathematics*, number 48 in Oxford Logic Guides, pages 91–114. Oxford University Press, 2005.

14 M. E. Maietti and G. Sambin. Why topology in the Minimalist Foundation must be pointfree. *Logic and Logical Philosophy*, 22(2):167–199, 2013.

15 P. Martin-Löf. *Notes on Constructive Mathematics.* Almqvist & Wiksell, 1970.

16 P. Martin-Löf. *Intuitionistic Type Theory. Notes by G. Sambin of a series of lectures given in Padua, June 1980.* Bibliopolis, Naples, 1984.

17 B. Nordström, K. Petersson, and J. Smith. *Programming in Martin Löf's Type Theory.* Clarendon Press, Oxford, 1990.

18 P. Odifreddi. *Classical recursion theory.*, volume 125 of *Studies in Logic and the Foundations of Mathematics.* North-Holland Publishing Co., 1989.

19 E. Palmgren. Bishop's set theory. Slides for lecture at the TYPES summer school, 2005.

20 A. S. Troelstra and D. van Dalen. Constructivism in mathematics, an introduction, vol. I. In *Studies in logic and the foundations of mathematics.* North-Holland, 1988.