Canonicity and decidability of equality for setoid type theory

István Donkó[1]*and Ambrus Kaposi[1]†

Eötvös Loránd University, Budapest, Hungary
{isti115,akaposi}@inf.elte.hu

Abstract

A proof assistant based on Setoid Type Theory would be practical for dealing with quotient inductive types. In order for such a system to be implemented in a proof assistant, we have to prove that it has decidability of equality. Our aim is to verify this property by constructing a syntactic translation from the syntax of Martin-Löf Type Theory and showing that this translation is injective.

1 Motivation

Type theory has proven to be an indispensable tool for precisely formalizing statements as well as constructing and validating proofs for them. In its current implementations handling some problems is quite cumbersome though. For example dealing with quotient types involves either lots of extra manual work, because the added equalities need to be eliminated manually (so called ”transport hell”) or the other option is to enable certain language features (e.g. rewrite rules in Agda [**?**]) which in turn deteriorate the computational properties of the type theory.

One possible alleviation of this problem could be basing proof assistants on Setoid Type Theory (SeTT, [**?**, **?**]), also called observational type theory ([**?**, **?**]), which would natively enable the usage of quotient inductive types. SeTT is based on the setoid model where the equality relation for any type can be specified arbitrarily.

2 Requirements

For SeTT to be usable for such purposes, it needs to have certain properties such as canonicity and decidable equality which is necessary for type checking. Instead of proving these properties using usual methods such as logical relations [**?**], we simplify the procedure using a model construction. We call the model construction *setoidification*. We use the variant of the model construction described in [**?**]. Any model of Martin-Löf type theory with strict propositions (MLTTP) can be turned into a model of SeTT. We aim to transport the necessary properties of the model of MLTTP to its setoidified version. In particular, we want to prove that the interpretation of the syntax of SeTT into the setoidified syntax of MLTTP is injective. Injectivity can also be called completeness: every equality that holds in the setoidification of MLTTP is reflected in the syntax of SeTT.

*Supported by the ÚNKP-21-3 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund.

†Ambrus Kaposi was supported by the “Application Domain Specific Highly Reliable IT Solutions” project which has been implemented with support from the National Research, Development and Innovation Fund of Hungary, financed under the Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme) funding scheme, and by Bolyai Scholarship BO/00659/19/3.

A term in the setoidified model t is a term in the original model $|t|$ together with a proof that it respect the equivalence relation belonging to the type of the term. We call the projection $|\_|$ the *evaluation* function.

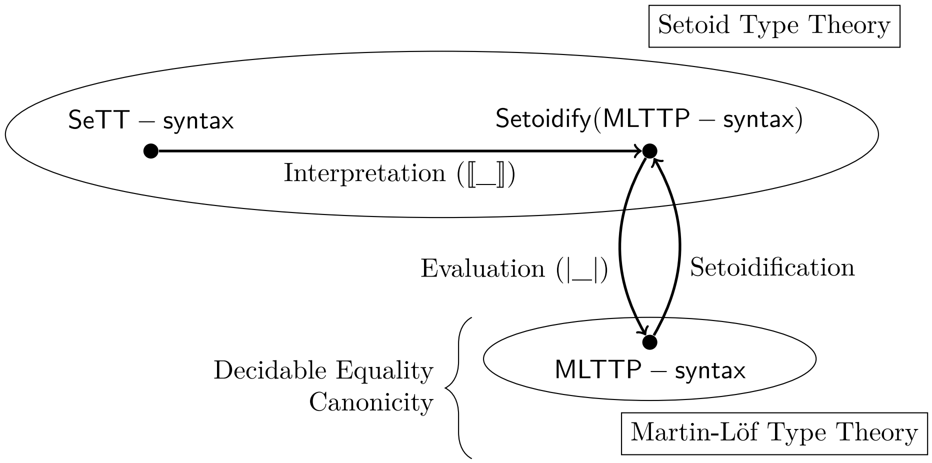

Figure 1: Setoidification illustrated

Let the $[\![\_]\!]$ operation be the interpretation from the syntax of SeTT to the setoidification of the syntax of MLTTP. We assume injectivity of the interpretation function, that is, for any t, t' terms in SeTT of the same type, $[\![t]\!] = [\![t']\!] \Rightarrow t = t'$.

- *Decidability of equality*
  Decidability of equality states that either $t = t'$ or $t \neq t'$ holds. Through interpretation and evaluation, $|[\![t]\!]| = |[\![t']\!]|$ is an equality in the base model. If we have decidable equality there, we have two cases:
  - $|[\![t]\!] = |[\![t']\!]|$ implies $[\![t]\!] = [\![t']\!]$ which implies $t = t'$ by injectivity
  - $|[\![t]\!]| \neq |[\![t']\!]|$ implies $t \neq t'$ by contradiction
- *Canonicity*
  Canonicity means that every closed term can be equated to one that is only built using the constructors of the given type. For example, the type Bool has two canonical forms in the base model:
  - $|[\![t]\!]| = \mathsf{false}_{\mathsf{MLTTP-syntax}}$ implies $t = \mathsf{false}_{\mathsf{SeTT-syntax}}$ by injectivity
  - $|[\![t]\!]| = \mathsf{true}_{\mathsf{MLTTP-syntax}}$ implies $t = \mathsf{true}_{\mathsf{SeTT-syntax}}$ by injectivity

We are in the process of proving injectivity of interpretation into setoidification following the analogous proof of injectivity of interpretation into termification [**?**].

References

[1] Agda Development Team. Agda - https://github.com/agda/agda.

[2] Thorsten Altenkirch, Simon Boulier, Ambrus Kaposi, Christian Sattler, and Filippo Sestini. Constructing a universe for the setoid model. In Stefan Kiefer and Christine Tasson, editors, *Foundations of Software Science and Computation Structures - 24th International Conference, FOSSACS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings*, volume 12650 of *Lecture Notes in Computer Science*, pages 1–21. Springer, 2021.

[3] Thorsten Altenkirch, Simon Boulier, Ambrus Kaposi, and Nicolas Tabareau. Setoid type theory—a syntactic translation. In Graham Hutton, editor, *Mathematics of Program Construction*, pages 155–196, Cham, 2019. Springer International Publishing.

[4] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. Observational equality, now! In Aaron Stump and Hongwei Xi, editors, *Proceedings of the ACM Workshop Programming Languages meets Program Verification, PLPV 2007, Freiburg, Germany, October 5, 2007*, pages 57–68. ACM, 2007.

[5] Ambrus Kaposi, András Kovács, and Nicolai Kraus. Shallow embedding of type theory is morally correct. In Graham Hutton, editor, *Mathematics of Program Construction - 13th International Conference, MPC 2019, Porto, Portugal, October 7-9, 2019, Proceedings*, volume 11825 of *Lecture Notes in Computer Science*, pages 329–365. Springer, 2019.

[6] Loïc Pujet and Nicolas Tabareau. Observational equality: now for good. *Proc. ACM Program. Lang.*, 6(POPL):1–27, 2022.