DutchCATS
Strength of Weak Type Theory
Amsterdam, 11-12 May 2023

Weak type theories: a conservativity result for homotopy elementary types.

Matteo Spadetto (University of Leeds)

Objective/propositional/weak type theory

In:

Benno van den Berg, Martijn den Besten, *Quadratic type checking for objective type theory*, 2021.

the authors consider a type theory without judgemental equalities and study the decidability of its derivability of a given term judgement. In this paper such a type theory is called objective type theory.

Objective/propositional/weak type theory

In:

Benno van den Berg, Martijn den Besten, *Quadratic type checking for objective type theory*, 2021.

the authors consider a type theory without judgemental equalities and study the decidability of its derivability of a given term judgement. In this paper such a type theory is called objective type theory.

The authors claim that such a type theory is sufficient for performing all of constructive mathematics and formalising most of the HoTT book.

The notion of conservativity

Suppose that we are given two dependent type theories T_1 and T_2 such that T_2 extends T_1.

One can ask whether the two theories prove the same *statements*.

The notion of conservativity

Suppose that we are given two dependent type theories T_1 and T_2 such that T_2 extends T_1.

One can ask whether the two theories prove the same *statements*.

This property corresponds to the following: whenever T_2 proves a term-judgement $\Gamma \vdash a : A$, then T_1 proves a term-judgement $\Gamma \vdash \tilde{a} : A$.

When this happens, we say that T_2 is *conservative* over T_1.

Hofmann's conservativity result

If T_2 is an extensional type theory and T_1 is an intensional type theory with some (fundamental) additional extensional assumptions, e.g.

$$x : A, p : x = x \vdash p = r(x)$$

then T_2 is conservative over T_1. The proof is contained in:

Martin Hofmann, *Conservativity of equality reflection over intensional type theory*, 1996.

Intensional identity types

Formation & Introduction rules.

$$\frac{\vdash A : \textsc{Type}}{\begin{array}{c} x, x' : A \vdash x = x' : \textsc{Type} \\ x : A \vdash r(x) : x = x \end{array}}$$

Path Elimination & Computation rules.

$$\frac{\begin{array}{c} \vdash A : \textsc{Type} \\ x, x' : A;\ p : x = x' \vdash C(x, x', p) : \textsc{Type} \\ x : A \vdash q(x) : C(x, x, r(x)) \end{array}}{\begin{array}{ll} x, x' : A;\ p : x = x' \vdash & J(q, x, x', p) : C(x, x', p) \\ x : A \vdash & J(q, x, x, r(x)) \equiv q(x) \end{array}}$$

Propositional identity types

Formation & Introduction rules.

$$\frac{\vdash A : \textsc{Type}}{\begin{array}{c} x, x' : A \vdash x = x' : \textsc{Type} \\ x : A \vdash r(x) : x = x \end{array}}$$

Path Elimination & Propositional Computation rules.

$$\frac{\begin{array}{c} \vdash A : \textsc{Type} \\ x, x' : A;\ p : x = x' \vdash C(x, x', p) : \textsc{Type} \\ x : A \vdash q(x) : C(x, x, r(x)) \end{array}}{\begin{array}{c} x, x' : A;\ p : x = x' \vdash J(q, x, x', p) : C(x, x', p) \\ x : A \vdash \qquad J(q, x, x, r(x)) \not\equiv q(x) \end{array}}$$

Propositional identity types

Formation & Introduction rules.

$$\frac{\vdash A : \textsc{Type}}{\begin{array}{c} x, x' : A \vdash x = x' : \textsc{Type} \\ x : A \vdash r(x) : x = x \end{array}}$$

Path Elimination & Propositional Computation rules.

$$\frac{\begin{array}{c} \vdash A : \textsc{Type} \\ x, x' : A;\ p : x = x' \vdash C(x, x', p) : \textsc{Type} \\ x : A \vdash q(x) : C(x, x, r(x)) \end{array}}{\begin{array}{c} x, x' : A;\ p : x = x' \vdash J(q, x, x', p) : C(x, x', p) \\ x : A \vdash H(q, x) : J(q, x, x, r(x)) = q(x) \end{array}}$$

Propositional identity types in the literature

Propositional identity types appear in:

- Thierry Coquand, Nils Danielsson, *Isomorphism is equality*, 2013.
- Marc Bezem, Thierry Coquand, Simon Huber, *A model of type theory in cubical sets*, 2014.
- Benno van den Berg, *Path categories and propositional identity types*, 2018.
- Benno van den Berg, Martijn den Besten, *Quadratic type checking for objective type theory*, 2021.

Dependent sum types

Formation and Introduction rules.

$$\frac{\begin{array}{c}\vdash A : \textsc{Type} \\ x : A \vdash B(x) : \textsc{Type}\end{array}}{\begin{array}{c}\vdash \Sigma_{x:A} B(x) : \textsc{Type} \\ x : A, y : B(x) \vdash \langle x, y \rangle : \Sigma_{x:A} B(x)\end{array}}$$

Elimination and Computation rules.

$$\frac{\begin{array}{c}\vdash A : \textsc{Type} \\ x : A \vdash B(x) : \textsc{Type} \\ u : \Sigma_{x:A} B(x) \vdash C(u) : \textsc{Type} \\ x : A;\ y : B(x) \vdash c(x, y) : C(\langle x, y \rangle)\end{array}}{\begin{array}{c}u : \Sigma_{x:A} B(x) \vdash \mathrm{split}(c, u) : C(u) \\ x : A;\ y : B(x) \vdash \qquad \mathrm{split}(c, \langle x, y \rangle) \equiv c(x, y)\end{array}}$$

Propositional dependent sum types

Formation and Introduction rules.

$$\frac{\vdash A : \textsc{Type} \qquad x : A \vdash B(x) : \textsc{Type}}{\vdash \Sigma_{x:A} B(x) : \textsc{Type} \qquad x : A, y : B(x) \vdash \langle x, y \rangle : \Sigma_{x:A} B(x)}$$

Elimination and Computation rules.

$$\frac{\vdash A : \textsc{Type} \qquad x : A \vdash B(x) : \textsc{Type} \qquad u : \Sigma_{x:A} B(x) \vdash C(u) : \textsc{Type} \qquad x : A;\ y : B(x) \vdash c(x, y) : C(\langle x, y \rangle)}{u : \Sigma_{x:A} B(x) \vdash \mathrm{split}(c, u) : C(u) \qquad x : A;\ y : B(x) \vdash \mathrm{split}(c, \langle x, y \rangle) \not\equiv c(x, y)}$$

Propositional dependent sum types

Formation and Introduction rules.

$$\frac{\vdash A : \textsc{Type} \qquad x : A \vdash B(x) : \textsc{Type}}{\vdash \Sigma_{x:A} B(x) : \textsc{Type} \qquad x : A, y : B(x) \vdash \langle x, y \rangle : \Sigma_{x:A} B(x)}$$

Elimination and Propositional Computation rules.

$$\frac{\vdash A : \textsc{Type} \qquad x : A \vdash B(x) : \textsc{Type} \qquad u : \Sigma_{x:A} B(x) \vdash C(u) : \textsc{Type} \qquad x : A;\ y : B(x) \vdash c(x, y) : C(\langle x, y \rangle)}{u : \Sigma_{x:A} B(x) \vdash \mathrm{split}(c, u) : C(u) \qquad x : A;\ y : B(x) \vdash \sigma(c, x, y) : \mathrm{split}(c, \langle x, y \rangle) = c(x, y)}$$

Aim of the talk

Let PTT be a dependent type theory with:

- propositional identity types,
- propositional dependent sum types,
- and propositional dependent product types.

Aim of the talk

Let PTT be a dependent type theory with:

- propositional identity types,
- propositional dependent sum types,
- and propositional dependent product types.

By adapting Hofmann's argument, we compare PTT to extensional type theories, looking for a conservativity result of such a type theory over PTT.

h-elementary types

Definition

The class of the h-elementary type-judgements is the smallest class $\mathcal{F}$ of type-judgements of PTT such that:

- atomic type-judgements $\gamma \vdash S : \textsc{Type}$ belongs to $\mathcal{F}$ if S is an h-set;
- the family $\mathcal{F}$ is closed up to applying the formation rules of $=$, Π and Σ.

h-elementary types

Definition

The class of the h-elementary type-judgements is the smallest class $\mathcal{F}$ of type-judgements of PTT such that:

- atomic type-judgements $\gamma \vdash S : \textsc{Type}$ belongs to $\mathcal{F}$ if S is an h-set;
- the family $\mathcal{F}$ is closed up to applying the formation rules of $=$, Π and Σ.

A type-judgement $\delta \vdash T(\delta) : \textsc{Type}$ of PTT is h-elementary if it belongs to this class.

Let ETT be the extensional type theory whose atomic types are the atomic h-sets of PTT.

Let $|\cdot| : \mathrm{PTT} \rightharpoonup \mathrm{ETT}$ be the interpretation of:

1. the h-elementary contexts of PTT,
2. the h-elementary type-judgements of PTT in h-elementary context,
3. the term-judgements of PTT in h-elementary type and context,

i.e. the h-elementary sub-theory of PTT, in ETT.

Theorem

Whenever $\gamma : \Gamma$ *is an h-elementary context of* PTT *and:*

$$\gamma \vdash A(\gamma) : \textsc{Type}$$

is an h-elementary type-judgement of PTT, *if* ETT *infers* $|\gamma| \vdash_{\texttt{ext}} a(|\gamma|) : |A(\gamma)|$ *then* PTT *infers:*

$$\gamma \vdash \tilde{a}(\gamma) : A(\gamma).$$

Theorem

Whenever $\gamma : \Gamma$ *is an h-elementary context of* PTT *and:*

$$\gamma \vdash A(\gamma) : \textsc{Type}$$

is an h-elementary type-judgement of PTT, *if* ETT *infers* $|\gamma| \vdash_{\mathtt{ext}} a(|\gamma|) : |A(\gamma)|$ *then* PTT *infers:*

$$\gamma \vdash \tilde{a}(\gamma) : A(\gamma).$$

If the only atomic types of PTT *are the empty type, the unit type, the type of booleans and the natural numbers type, then* ETT *is conservative over* PTT.

Categories with attributes...

A *category with attributes* $(\mathcal{C}, \textsc{tp}, -.-, P)$ consists of:

- A category $\mathcal{C}$ of *semantic contexts* Γ, Δ, ...
- A presheaf $\mathcal{C}^{\mathrm{op}} \xrightarrow{\textsc{tp}} \textsc{Set}$ of *semantics types* A, B... in some semantic context Γ
- A *semantic context extension* $\int \textsc{tp} \to \mathcal{C}$, denoted as $(\Gamma, A) \mapsto \Gamma.A$
- A cartesian natural family of display maps $\Gamma.A \xrightarrow{P_A} \Gamma$

The *semantic terms* of A are the sections $\Gamma \xrightarrow{a} \Gamma.A$ of $\Gamma.A \xrightarrow{P_A} \Gamma$.

...i.e. models of dependent type theories

Suppose that we are given a morphism $\Delta \xrightarrow{f} \Gamma$. Then we can define the mapping $a \mapsto a[f]$ as follows:

$$\begin{tikzcd}
\Delta \arrow[rr, "f"] \arrow[dr, dotted, "{a[f]}"'] & & \Gamma \arrow[dr, "a"] & \\
& \Delta.A[f] \arrow[rr, "f.A"] \arrow[d, "P_{A[f]}"'] \arrow[drr, phantom, very near start, "\lrcorner"] & & \Gamma.A \arrow[d, "P_A"] \\
& \Delta \arrow[rr, "f"] & & \Gamma
\end{tikzcd}$$

Idea of the proof

1. One can interpret judgements of a given dependent type theory into an appropriate category with attributes.

2. One can say when a category with attributes is endowed with (semantic) extensional/intensional/propositional id types.

Idea of the proof

1. One can interpret judgements of a given dependent type theory into an appropriate category with attributes.
2. One can say when a category with attributes is endowed with (semantic) extensional/intensional/propositional id types.
3. This provides a notion of *sound semantics* for the corresponding extensional/intensional/propositional type theory.
4. *Proof strategy.* Constructing a model M (according to this notion of semantics) of ETT, such that the interpretation of the h-elementary sub-theory of PTT in M is surjective (on contexts, types and terms).

Idea of the proof

In fact, let us assume that we found such a model M.

And let us assume that $\gamma \vdash A : \textsc{Type}$ is h-elementary in PTT and that $|\gamma| \vdash a : |A|$ in ETT.

Idea of the proof

In fact, let us assume that we found such a model M.

And let us assume that $\gamma \vdash A : \textsc{Type}$ is h-elementary in PTT and that $|\gamma| \vdash a : |A|$ in ETT. Then:

$$\begin{tikzcd}
|\gamma|^M \arrow[r, "a^M"] \arrow[dr, equal] & |\gamma|^M.|A|^M \arrow[d, "P_{|A|^M}"] \\
& |\gamma|^M
\end{tikzcd}$$

Idea of the proof

In fact, let us assume that we found such a model M.

And let us assume that $\gamma \vdash A : \textsc{Type}$ is h-elementary in PTT and that $|\gamma| \vdash a : |A|$ in ETT. Then:

$$\begin{tikzcd}
|\gamma|^M \arrow[r, "a^M"] \arrow[dr, equal] & |\gamma|^M.|A|^M \arrow[d, "P_{|A|^M}"] \\
& |\gamma|^M
\end{tikzcd}$$

Then there is some h-elementary term-judgement $\delta \vdash b : B$ in PTT whose interpretation in M coincides with the a^M.

By some initiality argument and by "the nature of M", one can assume that $\delta \equiv \gamma$ and $B \equiv A$, therefore $\gamma \vdash b : A$.

How to built M

We start from the syntax of PTT (which can be seen as a category with attributes itself) and we use it to build M.

How to built M

We start from the syntax of PTT (which can be seen as a category with attributes itself) and we use it to build M.

We must identify paths with reflexivities:

$$x, y : A \vdash p : x = y \quad \leftrightarrow \quad x : A \vdash r(x) : x = x$$

in a way that maintains the dependent type theoretic structure.

How to identify contexts

The identification of the contexts can be inductively defined via the notion of identification of the types:

if $\gamma : \Gamma$ and $\delta : \Delta$ in PTT are identified and, for some judgements:

$$\gamma \vdash A(\gamma) : \textsc{Type} \quad \text{and} \quad \delta \vdash B(\delta) : \textsc{Type}$$

it is the case that A and B are identified, then the extended contexts:

$$\gamma : \Gamma, x : A(\gamma) \quad \text{and} \quad \delta : \Delta, y : B(\delta)$$

will be identified as well.

Hence, we only need to define how to identify types.

How to identify types

We identify two types $\gamma \vdash A(\gamma) : \textsc{Type}$ and $\delta \vdash B(\delta) : \textsc{Type}$ if between them there is a canonical homotopy equivalence.

Canonical homotopy equivalences are defined inductively on the complexity of the types:

How to identify types

We identify two types $\gamma \vdash A(\gamma) : \textsc{Type}$ and $\delta \vdash B(\delta) : \textsc{Type}$ if between them there is a canonical homotopy equivalence.

Canonical homotopy equivalences are defined inductively on the complexity of the types:

- If $A(\gamma) \equiv B(\delta)$ is an atomic type, then the identity map is canonical.

How to identify types

We identify two types $\gamma \vdash A(\gamma) : \textsc{Type}$ and $\delta \vdash B(\delta) : \textsc{Type}$ if between them there is a canonical homotopy equivalence.

Canonical homotopy equivalences are defined inductively on the complexity of the types:

- If $A(\gamma) \equiv B(\delta)$ is an atomic type, then the identity map is canonical.
- If $A(\gamma) \equiv x_1 =_{A'(\gamma)} x_2$ and $B(\delta) \equiv y_1 =_{B'(\delta)} y_2$ and there is canonical $f\colon A'(\gamma) \to B'(\delta)$ together with:

  $$\delta \vdash q_1 \colon f(x_1) =_{B'(\delta)} y_1$$
  $$\delta \vdash q_2 \colon f(x_2) =_{B'(\delta)} y_2$$

  then the induced equivalence $p \mapsto q_1^{-1} \bullet f(p) \bullet q_2$ between $A(\gamma)$ and $B(\delta)$ is canonical.

How to identify types

We identify two types $\gamma \vdash A(\gamma) : \textsc{Type}$ and $\delta \vdash B(\delta) : \textsc{Type}$ if between them there is a canonical homotopy equivalence.

Canonical homotopy equivalences are defined inductively on the complexity of the types:

- If $A(\gamma) \equiv B(\delta)$ is an atomic type, then the identity map is canonical.
- If $A(\gamma) \equiv x_1 =_{A'(\gamma)} x_2$ and $B(\delta) \equiv y_1 =_{B'(\delta)} y_2$ and there is canonical $f \colon A'(\gamma) \to B'(\delta)$ together with:

  $$\delta \vdash q_1 \colon f(x_1) =_{B'(\delta)} y_1$$
  $$\delta \vdash q_2 \colon f(x_2) =_{B'(\delta)} y_2$$

  then the induced equivalence $p \mapsto q_1^{-1} \bullet f(p) \bullet q_2$ between $A(\gamma)$ and $B(\delta)$ is canonical.

  Hence, if $\vdash \overline{p} : x_1 = x_2$ then $p \mapsto \overline{p}^{-1} \bullet p \bullet r(x_2)$ is (essentially) canonical $x_1 = x_2 \to x_2 = x_2$ and identifies $\overline{p}$ with $r(x_2)$.
- Continue in the natural way for Π and Σ.

So what is M?

The semantic contexts of M are the equivalence classes of contexts of PTT modulo canonical context homotopy equivalences.

Two morphisms of contexts f and f' as in the diagram:

$$\begin{tikzcd}
\gamma \arrow[r, "f(\gamma)"] \arrow[d, "\mathsf{c}(\delta)"'] \arrow[dr, phantom, "//"] & \delta \arrow[d, "\mathsf{c}'(\delta)"] \\
\gamma' \arrow[r, "f'(\gamma')"'] & \delta'
\end{tikzcd}$$

represent the same morphism $[\gamma] \to [\delta]$ in M.

Issues & why h-elementary types

- If $[f] = [f']$ and $[g] = [g']$, that is:

$$
\begin{tikzcd}
\gamma \arrow[r, "\gamma \vdash f(\gamma)"] \arrow[d, "\mathsf{c}"'] \arrow[dr, phantom, "="] & \delta \arrow[r, "\delta \vdash g(\delta)"] \arrow[d, bend right, "\mathsf{c}'"'] \arrow[d, bend left, "\mathsf{c}''"] \arrow[dr, phantom, "="] & \omega \arrow[d, "\mathsf{c}'''"] \\
\gamma' \arrow[r, "\gamma' \vdash f'(\gamma')"'] & \delta' \arrow[r, "\delta' \vdash g'(\delta')"'] & \omega'
\end{tikzcd}
$$

then $[gf]$ is not necessarily $[g'f']$ because parallel canonical context homotopy equivalences are not necessarily homotopic.

Issues & why h-elementary types

- If $[f] = [f']$ and $[g] = [g']$, that is:

$$\begin{tikzcd}
\gamma \arrow[rr, "\gamma \vdash f(\gamma)"] \arrow[d, "\mathsf{c}"] & & \delta \arrow[rr, "\delta \vdash g(\delta)"] \arrow[d, bend right, "\mathsf{c}'"'] \arrow[d, bend left, "\mathsf{c}''"] & & \omega \arrow[d, "\mathsf{c}'''"] \\
\gamma' \arrow[rr, "\gamma' \vdash f'(\gamma')"'] \arrow[urr, phantom, "="] & & \delta' \arrow[rr, "\delta' \vdash g'(\delta')"'] \arrow[urr, phantom, "="] & & \omega'
\end{tikzcd}$$

then $[gf]$ is not necessarily $[g'f']$ because parallel canonical context homotopy equivalences are not necessarily homotopic.

Hence we need to allow two identity types to be identified only when they are mere propositions. We restrict ourselves to types with *h-propositional identities* and contexts with *h-propositional identities*.

Issues & why h-elementary types

- Even in this restriction, a naturality square:

$$\begin{tikzcd} \Delta.A[f] \arrow[r, "f.A"] \arrow[d, "P_{A[f]}"'] & \Gamma.A \arrow[d, "P_A"] \\ \Delta \arrow[r, "f"] & \Gamma \end{tikzcd}$$

  is a weak pullback but not necessarily a pullback.

Issues & why h-elementary types

- Even in this restriction, a naturality square:

$$\begin{tikzcd} \Delta.A[f] \arrow[r, "f.A"] \arrow[d, "P_{A[f]}"'] & \Gamma.A \arrow[d, "P_A"] \\ \Delta \arrow[r, "f"] & \Gamma \end{tikzcd}$$

  is a weak pullback but not necessarily a pullback.

  Hence we need to make a further restriction, i.e. we only allow types to be identified if they are h-elementary and in h-elementary context.

  In this case we actually obtain a category with attributes M that happens to be a model of ETT.

In order to have a look at the argument in detail:

Matteo Spadetto, *A conservativity result for homotopy elementary types in dependent type theory*, 2023.
arXiv 2303.05623