Harmony in Duality

30th International Conference on Types for Proofs and Programs — TYPES 2024

Henning Basold (Leiden University) Herman Geuvers (Radboud University Nijmegen)

11 June 2024

Universiteit Leiden
The Netherlands

Outline

What Is the Dual of Equality?

Dualising the usual equality gives ...

data Eq$_A$: A → A → **Ty where**
refl : (x : A) → ⊤ → **Eq** x x

codata InEq$_A$: A → A → **Ty where**
irefl : (x : A) → **InEq** x x → ⊥

Intuition

1. refl constructs term of **Eq** applied to twice the same argument
2. irefl observes only terms when **InEq** is applied twice to the same argument

Dualising the usual equality gives ...

data Eq$_A$: A → A → **Ty where**
refl : (x : A) → ⊤ → **Eq** x x

codata InEq$_A$: A → A → **Ty where**
irefl : (x : A) → **InEq** x x → ⊥

Intuition

1. refl constructs term of **Eq** applied to twice the same argument
2. irefl observes only terms when **InEq** is applied twice to the same argument

Dualising inductive equality gives classical inequality ☹

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types
- In cubical version of OTT via interval manipulation and induction on types

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types
- In cubical version of OTT via interval manipulation and induction on types
- Here: avoid intervals but
  1. define lifting of types to relations on terms

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \qquad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \qquad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types
- In cubical version of OTT via interval manipulation and induction on types
- Here: av
  1. defir

10:00 *Thorsten Altenkirch*, *Ambrus Kaposi*, *Michael Shulman* and *Elif Üsküplü*
Internal relational parametricity, without an interval (abstract)

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \qquad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \qquad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types
- In cubical version of OTT via interval manipulation and induction on types
- Here: avoid intervals but
  1. define lifting of types to relations on terms
  2. treat type equality via parameterised types and application to equality

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

Let us look somewhere else: Observational Type Theory

Rules for equality elimination via coercion and coherence[1]

$$\frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash s[Q\rangle : B} \qquad \frac{\Gamma \vdash Q : A \sim B \quad \Gamma \vdash s : A}{\Gamma \vdash \{s \,||\, Q\} : s \sim s[Q\rangle}$$

Proving type equality

- In OTT via definition of value equality and computation by induction on types
- In cubical version of OTT via interval manipulation and induction on types
- Here: avoid intervals but
  1. define lifting of types to relations on terms
  2. treat type equality via parameterised types and application to equality
  3. only equality proof principle is bisimilarity (coinductive!)

[1] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. "Observational Equality, Now!" In: *Proc. of PLPV '07. Workshop on Programming Languages Meets Program Verification*. ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6. DOI: 10.1145/1292597.1292608.

An Opportunity

Dualising coinductive equality

- Geuvers and Jacob have shown that apartness is the dual of bisimilarity[2]
- However, that work uses negation
- Traditional constructivists avoid that like the plague[3]

Reconciliation follows the same pattern as equality

1. define lifting of types to relations on terms
2. treat type apartness via parameterised types and application to equality
3. only apartness proof principle is inductive

[2] Herman Geuvers and Bart Jacobs. "Relating Apartness and Bisimulation". In: *Logical Methods in Computer Science* Volume 17, Issue 3 (July 30, 2021). ISSN: 1860-5974. DOI: 10.46298/lmcs-17(3:15)2021.

[3] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in Mathematics: An Introduction, Volume II*. Studies in Logic and the Foundations of Mathematics 123. North-Holland, 1988. 384 pp. ISBN: 0-444-70358-6.

Some Rules

Judgements

Well-formed type judgement

$\Delta \mid \Gamma_1 \vdash A : \Gamma_2 \twoheadrightarrow \mathbf{Ty}$ — A is a well-formed type with

- type variable context Δ
- object variable context Γ_1
- parameter context Γ_2

Well-formed term judgement

$\Gamma_1 \vdash t : \Gamma_2 \twoheadrightarrow A$ — t is well-formed term of type A with

- object variable context Γ_1
- parameter context Γ_2

Type formation

Parameter application and abstraction: simply typed λ-calculus for explicit substitutions

$$\frac{\Gamma_1 \vdash A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty} \qquad \Gamma_1 \vdash s : B}{\Gamma_1 \vdash A \mathbin{@} s : \Gamma_2[s/x] \twoheadrightarrow \mathbf{Ty}} \qquad \frac{\Gamma_1, x : B \vdash A : \Gamma_2 \twoheadrightarrow \mathbf{Ty}}{\Gamma_1 \vdash (x).\, A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty}}$$

[4] Henning Basold and Herman Geuvers. "Type Theory Based on Dependent Inductive and Coinductive Types". In: *Proceedings of LICS '16*. Logic In Computer Science. ACM, 2016, pp. 327–336. DOI: 10.1145/2933575.2934514. arXiv: 1605.02206; Henning Basold. "Mixed Inductive-Coinductive Reasoning: Types, Programs and Logic". PhD thesis. Radboud University, 2018. URL: https://hdl.handle.net/2066/190323.

Type formation

Parameter application and abstraction: simply typed λ-calculus for explicit substitutions

$$\frac{\Gamma_1 \vdash A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty} \qquad \Gamma_1 \vdash s : B}{\Gamma_1 \vdash A \mathbin{@} s : \Gamma_2[s/x] \twoheadrightarrow \mathbf{Ty}} \qquad \frac{\Gamma_1, x : B \vdash A : \Gamma_2 \twoheadrightarrow \mathbf{Ty}}{\Gamma_1 \vdash (x).\, A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty}}$$

Recursive types

$$\frac{\forall 1 \leq k \leq \left|\overrightarrow{A}\right|. \qquad \Delta, X : \Gamma \twoheadrightarrow \mathbf{Ty} \mid \Gamma \vdash A_k : \mathbf{Ty} \qquad \varrho \in \{\mu, \nu\}}{\Delta \mid \cdot \vdash \varrho(X : \Gamma \twoheadrightarrow \mathbf{Ty}\,; \overrightarrow{A}) : \Gamma \twoheadrightarrow \mathbf{Ty}} \textbf{(FP-Ty-}\varrho\textbf{)}$$

[4] Henning Basold and Herman Geuvers. "Type Theory Based on Dependent Inductive and Coinductive Types". In: *Proceedings of LICS '16*. Logic In Computer Science. ACM, 2016, pp. 327–336. DOI: 10.1145/2933575.2934514. arXiv: 1605.02206; Henning Basold. "Mixed Inductive-Coinductive Reasoning: Types, Programs and Logic". PhD thesis. Radboud University, 2018. URL: https://hdl.handle.net/2066/190323.

Type formation

Parameter application and abstraction: simply typed λ-calculus for explicit substitutions

$$\frac{\Gamma_1 \vdash A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty} \qquad \Gamma_1 \vdash s : B}{\Gamma_1 \vdash A \mathbin{@} s : \Gamma_2[s/x] \twoheadrightarrow \mathbf{Ty}} \qquad \frac{\Gamma_1, x : B \vdash A : \Gamma_2 \twoheadrightarrow \mathbf{Ty}}{\Gamma_1 \vdash (x).\, A : (x : B, \Gamma_2) \twoheadrightarrow \mathbf{Ty}}$$

Recursive types

$$\frac{\forall 1 \leq k \leq \left|\overrightarrow{A}\right|. \qquad \Delta, X : \Gamma \twoheadrightarrow \mathbf{Ty} \mid \Gamma \vdash A_k : \mathbf{Ty} \qquad \varrho \in \{\mu, \nu\}}{\Delta \mid \cdot \vdash \varrho(X : \Gamma \twoheadrightarrow \mathbf{Ty}\,; \overrightarrow{A}) : \Gamma \twoheadrightarrow \mathbf{Ty}} \textbf{(FP-Ty-}\varrho\textbf{)}$$

Expressiveness

- Sigma (inductive), Pi (coinductive) and usual recursive types can be expressed[4]
- The type **Eq** is not definable in as inductive type because constructors don't allow substitutions

[4] Henning Basold and Herman Geuvers. "Type Theory Based on Dependent Inductive and Coinductive Types". In: *Proceedings of LICS '16*. Logic In Computer Science. ACM, 2016, pp. 327–336. DOI: 10.1145/2933575.2934514. arXiv: 1605.02206; Henning Basold. "Mixed Inductive-Coinductive Reasoning: Types, Programs and Logic". PhD thesis. Radboud University, 2018. URL: https://hdl.handle.net/2066/190323.

Type Relations

Type relation judgement

$\Gamma \vdash R$ **TyRel** — R is a well-formed type relation in context Γ

Type Relations

Type relation judgement

$\Gamma \vdash R \ \ \mathbf{TyRel}$ — R is a well-formed type relation in context Γ

We have heterogeneous and type equality as in OTT ...

$$\frac{\Gamma \vdash A, B : \mathbf{Ty} \qquad \Gamma \vdash s : A \qquad \Gamma \vdash t : B}{\Gamma \vdash s \sim t : \mathbf{Ty}} \qquad \frac{\Gamma_1 \vdash A : \Gamma_2 \to \mathbf{Ty} \qquad \Gamma_1 \vdash B : \Gamma_2 \to \mathbf{Ty}}{\Gamma \vdash A \sim B \ \ \mathbf{TyRel}}$$

Type Relations

Type relation judgement

$\Gamma \vdash R\ \ \mathbf{TyRel}$ — R is a well-formed type relation in context Γ

We have heterogeneous and type equality as in OTT ...

$$\frac{\Gamma \vdash A, B : \mathbf{Ty} \qquad \Gamma \vdash s : A \qquad \Gamma \vdash t : B}{\Gamma \vdash s \sim t : \mathbf{Ty}} \qquad \frac{\Gamma_1 \vdash A : \Gamma_2 \to \mathbf{Ty} \qquad \Gamma_1 \vdash B : \Gamma_2 \to \mathbf{Ty}}{\Gamma \vdash A \sim B\ \ \mathbf{TyRel}}$$

... heterogeneous apartness ...

$$\frac{\Gamma \vdash A, B : \mathbf{Ty} \qquad \Gamma \vdash s : A \qquad \Gamma \vdash t : B}{\Gamma \vdash s \# t : \mathbf{Ty}}$$

Type Relations

Type relation judgement

$\Gamma \vdash R\ \ \mathbf{TyRel}$ — R is a well-formed type relation in context Γ

We have heterogeneous and type equality as in OTT ...

$$\frac{\Gamma \vdash A, B : \mathbf{Ty} \qquad \Gamma \vdash s : A \qquad \Gamma \vdash t : B}{\Gamma \vdash s \sim t : \mathbf{Ty}} \qquad \frac{\Gamma_1 \vdash A : \Gamma_2 \to \mathbf{Ty} \qquad \Gamma_1 \vdash B : \Gamma_2 \to \mathbf{Ty}}{\Gamma \vdash A \sim B\ \ \mathbf{TyRel}}$$

... heterogeneous apartness ...

$$\frac{\Gamma \vdash A, B : \mathbf{Ty} \qquad \Gamma \vdash s : A \qquad \Gamma \vdash t : B}{\Gamma \vdash s \# t : \mathbf{Ty}}$$

... and one could use type apartness

$$\frac{\Gamma_1 \vdash A : \Gamma_2 \to \mathbf{Ty} \qquad \Gamma_1 \vdash B : \Gamma_2 \to \mathbf{Ty}}{\Gamma \vdash A \# B\ \ \mathbf{TyRel}}$$

How do we prove type relations?

Following our category theoretical nose

1. Define equality and apartness liftings of types to relations by induction on types
2. Use to devise proof principles for relations on terms using these liftings
3. Push term relation to type level

How do we prove type relations?

Following our category theoretical nose

1. Define equality and apartness liftings of types to relations by induction on types
2. Use to devise proof principles for relations on terms using these liftings
3. Push term relation to type level

Equality lifting

$$\frac{X : \Gamma \nrightarrow \mathbf{Ty} \mid \Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad \Gamma \vdash R : (\Gamma, x : B, y : B) \nrightarrow \mathbf{Ty}}{\vdash A^\flat(R) : (\Gamma, x : A[B/X], y : A[B/X]) \nrightarrow \mathbf{Ty}}$$

How do we prove type relations?

Following our category theoretical nose

1. Define equality and apartness liftings of types to relations by induction on types
2. Use to devise proof principles for relations on terms using these liftings
3. Push term relation to type level

Equality lifting

$$\frac{X : \Gamma \nrightarrow \mathbf{Ty} \mid \Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad \Gamma \vdash R : (\Gamma, x : B, y : B) \nrightarrow \mathbf{Ty}}{\vdash A^{\flat}(R) : (\Gamma, x : A[B/X], y : A[B/X]) \nrightarrow \mathbf{Ty}}$$

Apartness lifting

$$\frac{X : \Gamma \nrightarrow \mathbf{Ty} \mid \Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad \Gamma \vdash R : (\Gamma, x : B, y : B) \nrightarrow \mathbf{Ty}}{\vdash A^{\sharp}(R) : (\Gamma, x : A[B/X], y : A[B/X]) \nrightarrow \mathbf{Ty}}$$

Back to equality

Equality is coinductive

$$\frac{\Gamma_1 \vdash R : (x : \nu, y : \nu) \multimap \mathbf{Ty} \qquad x_k : \nu, y_k : \nu, z_k : R \mathbin{@} x_k \mathbin{@} y_k \vdash g_k : A_k^\flat(R)[\xi_k \mathbin{@} x_k / x, \xi_k \mathbin{@} y_k / y]}{\Gamma \vdash \mathrm{coind}\ \overrightarrow{(x_k, y_k, z_k).\, g_k} : (x : \nu, y : \nu, z : R \mathbin{@} x \mathbin{@} y) \multimap x \sim y}$$

Back to equality

Equality is coinductive

$$\frac{\Gamma_1 \vdash R : (x : \nu, y : \nu) \to \mathbf{Ty} \qquad x_k : \nu, y_k : \nu, z_k : R \mathbin{@} x_k \mathbin{@} y_k \vdash g_k : A_k^\flat(R)[\xi_k \mathbin{@} x_k / x, \xi_k \mathbin{@} y_k / y]}{\Gamma \vdash \mathrm{coind}\ \overrightarrow{(x_k, y_k, z_k).\, g_k} : (x : \nu, y : \nu, z : R \mathbin{@} x \mathbin{@} y) \to x \sim y}$$

Type equality via application – no recursive elimination

$$\frac{\Gamma_1 \vdash A : \Gamma_2 \to \mathbf{Ty} \qquad \Gamma_1 \vdash B : \Gamma_2 \to \mathbf{Ty} \qquad A \equiv B}{\Gamma \vdash \mathrm{Refl} : A \sim B}$$

$$\frac{\Gamma_1 \vdash A : (x : B, \Gamma_2) \to \mathbf{Ty} \qquad \Gamma_1 \vdash s, t : B \qquad \Gamma_1 \vdash p : s \sim t}{\Gamma_1 \vdash A \mathbin{@} p : A \mathbin{@} s \sim A \mathbin{@} t}$$

Using apartness

Apartness elimination

$$\frac{\Gamma \vdash A : \mathbf{Ty} \quad \Gamma \vdash B : \mathbf{Ty} \quad s \equiv t \quad \Gamma \vdash p : s \# t \quad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

$$\frac{\Gamma \vdash P : (x : A) \twoheadrightarrow \mathbf{Ty} \quad \Gamma \vdash s, t : A \quad \Gamma, y : s \# t \vdash p_1 : C \quad \Gamma, z : P \mathbin{@} t \vdash p_2 : C \quad \Gamma \vdash q : P \mathbin{@} s}{\Gamma \vdash \mathrm{extr}(y.p_1, z.p_2)\, q : C}$$

[5] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in Mathematics: An Introduction, Volume II*. Studies in Logic and the Foundations of Mathematics 123. North-Holland, 1988. 384 pp. ISBN: 0-444-70358-6.

Using apartness

Apartness elimination

$$\frac{\Gamma \vdash A : \mathbf{Ty} \quad \Gamma \vdash B : \mathbf{Ty} \quad s \equiv t \quad \Gamma \vdash p : s \# t \quad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

$$\frac{\Gamma \vdash P : (x : A) \twoheadrightarrow \mathbf{Ty} \quad \Gamma \vdash s, t : A \quad \Gamma, y : s \# t \vdash p_1 : C \quad \Gamma, z : P \mathbin{@} t \vdash p_2 : C \quad \Gamma \vdash q : P \mathbin{@} s}{\Gamma \vdash \mathrm{extr}(y.p_1, z.p_2)\, q : C}$$

Intuition

1. Apartness contradicts equality
2. Every predicate is strongly extensional[5]: $P\, s \to s \# t \vee P\, t$

[5] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in Mathematics: An Introduction, Volume II*. Studies in Logic and the Foundations of Mathematics 123. North-Holland, 1988. 384 pp. ISBN: 0-444-70358-6.

What can we prove?

1. Transitivity and symmetry of equality
2. Symmetry and separation (if $s \# t$ then $s \# r$ or $r \# t$)
3. Function and coinductive element extensionality with respect to $\sim$
4. Aparteness of coinductive elements as inductive witnesses

Example (Strong extensionality of functions)

Given

$$f : A \to B \quad \text{and} \quad s, t : A \quad \text{and} \quad p : f\, s \# f\, t$$

Set

$$Q = (x).\, f\, x \# f\, t \quad : \quad (x : A) \to \mathbf{Ty} \qquad \begin{array}{l} \Longrightarrow\ Q \mathbin{@} s \equiv f\, s \# f\, t \\ \Longrightarrow\ Q \mathbin{@} t \equiv f\, t \# f\, t \end{array}$$

$$\frac{\Gamma, y : s \# t \vdash y : s \# t \qquad \Gamma, z : Q \mathbin{@} t \vdash \#\mathrm{elim}\, z : s \# t \qquad \Gamma \vdash p : Q \mathbin{@} s}{\Gamma \vdash \mathrm{extr}(y.y, z.p_2)\, p : s \# t}$$

Harmony at Last?

Some harmony and some cognitive dissonance

- The duality works well on terms

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd
- Type apartness as dual to type equality useful?

$$\frac{\Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad A \equiv B \qquad \Gamma \vdash Q : A \# B \qquad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd
- Type apartness as dual to type equality useful?

$$\frac{\Gamma \vdash A : \mathbf{Ty} \quad \Gamma \vdash B : \mathbf{Ty} \quad A \equiv B \quad \Gamma \vdash Q : A \# B \quad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

- Monotonicity witnesses for relation liftings (extending Ralph Matthes work from recursors)?

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd
- Type apartness as dual to type equality useful?

$$\frac{\Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad A \equiv B \qquad \Gamma \vdash Q : A \# B \qquad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

- Monotonicity witnesses for relation liftings (extending Ralph Matthes work from recursors)?
- More generally: relations via double categorical language?

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd
- Type apartness as dual to type equality useful?

$$\frac{\Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad A \equiv B \qquad \Gamma \vdash Q : A \# B \qquad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

- Monotonicity witnesses for relation liftings (extending Ralph Matthes work from recursors)?
- More generally: relations via double categorical language?

Some harmony and some cognitive dissonance

- The duality works well on terms
- Not so nice on types: how to make the elimination of type equality and apartness proper dual?
- The heterogeneous equality and apartness are odd
- Type apartness as dual to type equality useful?

$$\frac{\Gamma \vdash A : \mathbf{Ty} \qquad \Gamma \vdash B : \mathbf{Ty} \qquad A \equiv B \qquad \Gamma \vdash Q : A \# B \qquad \Gamma \vdash C : \mathbf{Ty}}{\Gamma \vdash \#\mathrm{elim}\, Q : C}$$

- Monotonicity witnesses for relation liftings (extending Ralph Matthes work from recursors)?
- More generally: relations via double categorical language?

Thank You!