Elaborating Inductive Definitions and Course-of-Values Induction in Cedille

Anonymous Author(s)

October 21, 2019

Contents

0 Figures and Definitions

0.1 Syntax

The reader is refered to [Stu18] by Aaron Stump for more information about the standard fragment of Cedille that does not include datatypes. In Figures 1 and 2, the new language constructs introduced by the datatype system are $\boxed{\text{boxed}}$.

a, u, x, y, z	term variables
X, Y, Z, R	type variables
$\boxed{c}$	constructors
$\boxed{D}$	datatypes

Figure 1: Identifiers

p	$::=$	x	variables
		$\lambda\, u.\, p$	functions
		c	constructors
		$p\ p'$	applications
		$\boxed{\mu\ u.\ p\ \{\ c_i\ \overline{a_i} \to p_i\}_{i=1..n}}$	recursive definitions
		$\boxed{\mu'\ p\ \{c_i\ \overline{a_i} \to p_i\}_{i=1..n}}$	case analysis
$\overline{a}$	$::=$	$\boxed{\emptyset \mid a\ \overline{a}}$	

Figure 2: Untyped terms

The construct $\text{IndEl}[D, R, \Delta, \Theta, \mathcal{E}]$ of Figure 4 is explained in Section 5.1. We also make implicit reference to the following judgment form

- $\Gamma \vdash t : T$, $\Gamma \vdash T : K$, and $\Gamma \vdash K$ indicating term t has type T, type T has kind K, and kind K is well-formed, in the fragment of Cedille *without* datatypes, described by the above reference

Kinds K	::=	$\star$	the kind of types that classify terms
		$\Pi\, X : K.\, K'$	product over types
		$\Pi\, x : T.\, K$	product over terms
Types S, T, P	::=	X	type variables
		$\Pi\, x : S.\, T$	product over terms
		$\forall\, x : S.\, T$	implicit product over terms
		$\forall\, X : K.\, T$	implicit product over types
		$\lambda\, x : S.\, T$	term-to-type function
		$\lambda\, X : K.\, T$	type-to-type function
		$T\ t$	type-to-term application
		$T\ \cdot S$	type-to-type application
		$\iota\, x : T.\, T'$	dependent intersection
		$\{p_1 \simeq p_2\}$	equality of untyped terms
		D	datatypes
Terms s, t	::=	x	variables
		$\lambda\, x.\, t$	term abstraction
		$\Lambda\, x.\, t$	erased term abstraction
		$\Lambda\, X.\, t$	type abstraction
		$t\ s$	term application
		$t\ \text{-}s$	erased term application
		$t \cdot T$	type application
		$[t, s]$	intro dependent intersection
		$t.1$	dep. intersection left projection
		$t.2$	dep. intersection right projection
		β	reflexivity of equality
		$\rho\ t\ @x.T\ -s$	rewrite by equality
		$\varphi\ t - t_1\ \{t_2\}$	cast by equality
		$\delta\ T - t$	anything by absurd equality
		c	data constructors
		$\mu\ x.\ t\ @P\ \{\ c_i\ \overline{a_i}\}_{i=1..n}$	recursive def. over datatype
		$\mu'\texttt{<}x\texttt{>}\ t\ @P\ \{\ c_i\ \overline{a_i}\}_{i=1..n}$	case analysis over datatype
Argument Sequence $\overline{s}$	::=	$\emptyset \mid s\ \overline{s} \mid \cdot S\ \overline{s} \mid \text{-}s\ \overline{s}$	for constructor patterns and applications

Figure 3: Syntax for Cedille kinds, types, terms

Typing contexts Γ ::= $\emptyset \mid \Gamma, x : T \mid \Gamma, X : K \mid \Gamma, \mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}]$

Figure 4: Contexts

0.2 Erasure

$$
\begin{array}{lcl}
|x| & = & x \\
|\star| & = & \star \\
|\{t \simeq t'\}|| & = & \{|t| \simeq |t'|\} \\
|\beta| & = & \lambda\, x.\, x \\
|\delta\ T - t| & = & |t| \\
|\varphi\ t\ \text{-}\ t'\ \{t''\}| & = & |t''| \\
|\rho\ t'\ @\ x.T\ \text{-}\ t| & = & |t| \\
|\iota\, x{:}T.\, T'| & = & \iota\, x{:}|T|.\, |T'| \\
|[t, t']| & = & |t| \\
|t.1| & = & |t| \\
|t.2| & = & |t| \\
|\Pi\, x{:}T.\, T'| & = & \Pi\, x{:}|T|.\, |T'| \\
|\lambda\, x.\, t| & = & \lambda\, x.\, |t| \\
|t\ t'| & = & |t|\ |t'| \\
|\forall\, x{:}T.\, T'| & = & \forall\, x{:}|T|.\, |T'| \\
|\Lambda\, x{:}T.\, t| & = & |t| \\
|t\ \text{-}t'| & = & |t| \\
|\forall\, X{:}K.\, T| & = & \forall\, X{:}|K|.\, |T| \\
|\Lambda\, X{:}K.\, t| & = & |t| \\
|t\ \cdot T| & = & |t| \\
|\Pi\, x{:}T.\, K| & = & \Pi\, x{:}|T|.\, |K| \\
|\lambda\, x{:}T.\, T'| & = & \lambda\, x{:}|T|.\, |T'| \\
|T\ t| & = & |T|\ |t| \\
|\Pi\, X{:}K.\, K'| & = & \Pi\, X{:}|K|.\, |K'| \\
|\lambda\, X{:}K.\, T| & = & \lambda\, X{:}|K|.\, |T| \\
|T \cdot T'| & = & |T| \cdot |T'| \\
|c| & = & c \\
|\mu\ u.\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}| & = & \mu\ u.\ |t|\ \{\ c_i\ \overline{|a_i|} \to |t_i|\}_{i=1..n} \\
|\mu'\texttt{<}u\texttt{>}\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}| & = & \mu'\ |t|\ \{c_i\ \overline{|a_i|} \to |t_i|\}_{i=1..n}
\end{array}
$$

$$
\begin{array}{lcl}
|\varnothing| & = & \varnothing \\
|s\ \overline{s}| & = & |s|\ |\overline{s}| \\
|\text{-}s\ \overline{s}| & = & |\overline{s}| \\
|\cdot\, S\ \overline{s}| & = & |\overline{s}|
\end{array}
$$

Figure 5: Erasure for annotated terms

0.3 Operational Semantics

Previous iterations of this work included results showing how to achieve zero-cost data and program reuse (as in [DFS18]) by a simple extension to definitional equality for datatype constructors. This was struck from the current submission due to space limitations, but we keep these results in this proof appendix anyway, because they are neat.

$$\begin{array}{rcl} |\ \mu'\texttt{<}is\texttt{>}\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}\ | & = & \mu'\ |t|\ \{c_i\ |\overline{a_i}| \to |t_i|\}_{i=1..n} \\ |\ \mu\ ih.\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}\ | & = & \mu\ ih.\ |t|\ \{\ c_i|\overline{a_i}| \to |t_i|\}_{i=1..n} \end{array}$$

$$\frac{1 \le j \le n \quad \#\overline{s} = \#\overline{a_j}}{\mu'\ (c_j\ \overline{s})\ \{c_i\ \overline{a_i} \to t_i\}_{i=1..n} \rightsquigarrow \overline{[s/a_j]}t_j} \qquad \frac{1 \le j \le n \quad \#\overline{s} = \#\overline{a_j} \quad r = \lambda\, x.\, \mu\ ih.\ x\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}}{\mu\ ih.\ (c_j\ \overline{s})\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n} \rightsquigarrow \overline{[s/a_j]}[r/ih]t_j}$$

Figure 6: Erasure and reduction for μ and μ'

$$\frac{\begin{array}{cc} \mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma & c_j : \overset{\Pi}{\forall}\ \overline{a_j : A_j}.\ D \in \Delta \\ \mathrm{IndEl}[D', R, \Delta', \Theta', \mathcal{E}'] \in \Gamma & {c_k}' : \overset{\Pi}{\forall}\ \overline{a_k : A_k}'.\ D' \in \Delta' \\ j = k, \#\Delta = \#\Delta' & \#|\overline{a_j}| = \#|\overline{a_k}| \end{array}}{\Gamma \vdash c_j \cong {c_k}'}$$

Figure 7: Extension of definitional equality for data declarations

In Figure 6, a metavariable c denotes a datatype constructor, $\overline{s}$ a sequence of type and (mixed-erasure) term arguments, $\overline{a}$ a sequence of type and (mixed-erasure) term variables bound by pattern guards, $\#\overline{s}$ the length of $\overline{s}$, $\{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}$ a collection of n branches guarded by patterns $c_i\ \overline{a_i}$ with bodies t_i, $|\overline{a}|$ the erasure of type and erased-term variables in the sequence $\overline{a}$, and $\overline{[s/a]}$ the simultaneous and capture-avoiding substitution of terms and types $\overline{s}$ for variables $\overline{a}$.

The convertibility relation $\cong$ for types is the relation described in [Stu18] with term convertibility augmented with these rules.

0.4 Generic Interface

We shall call Γ^{G} the typing context containing all of the definitions for the generic derivation of inductive for λ-encoded datatypes. This includes the definitions given in Figures 8 and 9.

(a) Casts, positivity, and type fixpoints

```
Cast: ⋆ → ⋆ → ⋆ = λ A: ⋆. λ B: ⋆. Σ f: A → B. {f ≃ λ x. x}.

intrCast: ∀ A: ⋆. ∀ B: ⋆. Π g: A → B. (Π x: A. {g x ≃ x}) → Cast ·A ·B
= Λ A. Λ B. λ g. λ e. (λ x. φ (e x) - (g x) {x} , β).

elimCast: ∀ A: ⋆. ∀ B: ⋆. ∀ _: Cast ·A ·B. A → B
= Λ A. Λ B. Λ c. φ (π2 c) - (π1 c) {λ x. x}.
_ : {elimCast ≃ λ x. x} = β.

Mono: (⋆ → ⋆) → ⋆ = λ F: ⋆ → ⋆. ∀ X: ⋆. ∀ Y: ⋆. Cast ·X ·Y → Cast ·(F ·X) ·(F ·Y).

intrMono: ∀ F: ⋆ → ⋆. (∀ X: ⋆. ∀ Y: ⋆. Cast ·X ·Y → Cast ·(F ·X) ·(F ·Y)) → Mono ·F
= Λ F. λ im. im.

elimMono: ∀ F: ⋆ → ⋆. ∀ _: Mono ·F. ∀ A: ⋆. ∀ B: ⋆. ∀ _: Cast ·A ·B. F ·A → F ·B
= Λ F. Λ im. Λ A. Λ B. Λ c. elimCast -(im c).
_ : {elimMono ≃ λ x. x} = β.

Fix: Π F: ⋆ → ⋆. Mono ·F → ⋆ = <..>
in:  ∀ F: ⋆ → ⋆. ∀ im: Mono ·F. F ·(Fix ·F im) → Fix ·F im = <..>
out: ∀ F: ⋆ → ⋆. ∀ im: Mono ·F. Fix ·F im → F ·(Fix ·F im) = <..>
```

(b) Generic constructs for λ-encoded inductive types

```
module GenericCoV (F: ⋆ → ⋆) {im: Mono ·F}

D: ⋆ = Fix ·F im.

PrfAlg: (D → ⋆) → ⋆
= λ P: D → ⋆. ∀ R: ⋆. ∀ c: Cast ·R ·D. Π o: R → F ·R. ∀ oeq: {o ≃ out}.
  (Π r: R. P (elimCast -c r)) → Π xs: F ·R. P (in (elimMono -im -c xs)).

induction: ∀ P: D → ⋆. PrfAlg ·P → Π x: D. P x = <..>

inductionComp: ∀ P: D → ⋆. Π alg: PrfAlg ·P. Π xs: F ·D.
  {induction alg (in xs) ≃ alg out (induction alg) xs}
= Λ P. λ alg. λ xs. β.

lambek1: Π xs: F ·D. {xs ≃ out (in xs)} = β.
lambek2: Π x: D. {x ≃ in (out x)} = induction ·P (Λ R. Λ c. λ o. Λ eq. λ ih. λ xs. β)
```

Figure 8: Generic framework

```
module DataInterface (F: ⋆ → ⋆) {im: Mono · F}.
import GenericCoV ·F -im.

View: Π A: ⋆. Π a: A. ⋆ → ⋆ = λ A: ⋆. λ a: A. λ B: ⋆. Σ b: B. {a ≃ b}.

intrView: ∀ A: ⋆. ∀ B: ⋆. ∀ a: A. Π b: B. {b ≃ a} → View ·A a ·B
= Λ A. Λ B. Λ x. λ y. λ eq. (y , eq).

elimView: ∀ A: ⋆. ∀ B: ⋆. Π a: A. ∀ _: View ·A a ·B. B
= Λ A. Λ B. λ a. Λ v. φ (π2 v) - (π1 v) {a}.
_ : {elimView ≃ λ x. x} = β.

IsD: ⋆ → ⋆ = λ R: ⋆. (Cast ·R ·D) × (View ·(D → F ·D) out ·(R → F ·R)).
isD: IsD ·D = (intrCast (λ x. x) (λ _. β) , intrView -out out β).
toD: ∀ R: ⋆. ∀ _: IsD ·R. R → D = Λ R. Λ is. λ r. elimCast -(π1 is) r.
toFD: ∀ R: ⋆. ∀ _: IsD ·R. F ·R → F ·D = Λ R. Λ is. λ fd. elimMono -pos -(π1 is) fd.

ByCases: (D → ⋆) → Π R: ⋆. IsD ·R → ⋆
= λ P: D → ⋆. λ R: ⋆. λ is: IsD ·R. Π fr: F ·R. P (in (toFD -is fr)).

mu': ∀ R: ⋆. ∀ is: IsD ·R. Π r: R. ∀ P: D → ⋆. ByCases ·P ·R is → P (toD -is r)
= Λ R. Λ is. λ r. Λ P. λ case. ρ (lambek2 (toD -is r)) - case (elimView out -(π2 is) r)

ByInd: (D → ⋆) → ⋆
= λ P: D → ⋆. ∀ R: ⋆. ∀ is: IsD ·R. (Π r: R. P (toD -is r)) → ByCases ·P ·R is.

mu: Π d: D. ∀ P: D → ⋆. ByInd · P → P d
= λ d. Λ P. λ ind. induction ·P (
  Λ R. Λ c. λ o. λ oeq. λ ih. λ fr. ind -(c , intrView -out o oeq) ih fr) d
```

Figure 9: Interface for datatype elaborator

We conclude by stating some requirements needed to show *value-preservation* (Theorem 6.1) and the *termination guarantee* (Theorem 6.3) of our elaborator that the generic framework satisfies.

Requirement 1. *Definitions* ***in*** *and* ***out*** *are mutual inverses. Furthermore, there is a constant bound such that for all* $xs : F \cdot D$*, expression* ***out*** $(\textbf{\textit{in}}\ xs)$ *β-reduces to* xs *in a number of steps within that bound.*

Requirement 2. *For every untyped λ-expression* a*, there exists some term* t *such that* ***induction*** $a \rightsquigarrow^* t$ *and that* ***induction*** $a\ (\textbf{\textit{in}}\ xs) \rightsquigarrow^* a$ ***out*** $t\ xs$*, for all terms* xs*.*

Requirement 3. *For closed definitions of* $F : \star \to \star$ *and* $im : \textbf{\textit{Mono}} \cdot F$*, there exists some closed term* t' *of type* $\textbf{\textit{Fix}} \cdot F\ im \to \Pi\, x : T_1.\, T_2$ *(for some* T_1 *and* T_2*) that erases to* $\lambda\, x.\, x$*.*

The first part of Requirement 1 is known as *Lambek's lemma* [Lam68]. Requirement 2 expresses the *cancellation law* for the initial Mendler-style CoV F-algebra (phrased differently: `induction` computes as a course-of-values recursor for data). Requirement 3 simply states that the elaborations of datatypes *must be functional*. All three requirements are satisfied by the library provided by [FDJS18].

0.5 Elaboration Rules

(a) Datatype declarations

```
data Bool: ⋆ = tt: Bool  | ff: Bool.
data Nat:  ⋆ = zero: Nat | suc: Nat → Nat.
data List (A: ⋆): ⋆
= nil: List | cons: A → List → List.
```

(b) Functions

```
pred: Nat → Nat = λ n. μ' n {zero → n | suc n' → n'}.
add: Nat → Nat → Nat
= λ m. λ n. μ addN. m {zero → n | suc m' → suc (addN m')}.
```

Figure 10: Example datatype declarations and functions

Notation In this section we give a formal description of the elaboration of non-indexed datatypes in Cedille to λ-encodings in Cedille 1.0.0 (which lacks datatypes); this is also the scope of the accompanying proof appendix. A declaration of datatype D of kind $\star$ is written $\mathrm{Ind}[D, R, \Delta]$, where

- R is a fresh type-variable of kind $\star$ whose scope is Δ
- Δ is the context of constructors associated with their type signatures such that all recursive occurrences of D in the types of constructor arguments in the surface language have been replaced by R

For example, the declaration of `Nat` in Figure 10 translates to

$$\mathrm{Ind}[\texttt{Nat}, R, \begin{array}{lcl} \texttt{zero} & : & \texttt{Nat} \\ \texttt{suc} & : & R \to \texttt{Nat} \end{array}]$$

We write $\Gamma \vdash \mathrm{Ind}[D, R, \Delta]$ *wf* to indicate that, for every i ranging from 1 to the number of constructors in Δ (written $i = 1..\#\Delta$), the ith constructor c_i in Δ has a type of the form $\overset{\Pi}{\forall}\overline{a_i : A_i}.D$ (indicating the mixed-erasure quantification over the dependent telescope of terms and types $\overline{a_i : A_i}$) that is well-kinded in context Γ extended by variables $R : \star$ and $D : \star$, and furthermore all occurrences of type D in the constructor arguments of the surface declaration have been replaced by R. Notation $\overset{\lambda}{\Lambda}\ \overline{a_i}.t$ and $t\ \overline{a_i}$ indicates resp. a term-level abstraction and application over this telescope respecting the erasures and classifiers over which the variables were quantified. This convention generalizes to the sequence of term and type expressions $\overline{s_i}$, as in $P\ (c_i\ \overline{s_i})$, when indicated that these are produced from type and kind coercions of $\overline{a_i}$.

By convention, judgments with the hooked arrow $\Gamma \vdash t : T \hookrightarrow t'$ are elaboration rules, written $\Gamma \vdash t : T \hookrightarrow \_$ when we need only that t is well-typed. Judgments without hooked arrows $\Gamma \vdash t : T$ and $\Gamma \vdash T : K$ indicate typing and kinding in Cedille 1.0.0. Some inference rules have premises in the form $(\Gamma \vdash \overset{\Pi}{\forall}\overline{a_i : A_i}.T : \star \hookrightarrow \_)_{i=1..\#\Delta}$, accompanied by a premise $(c_i : \overset{\Pi}{\forall}\overline{a_i : A_i}.\ D \in \Delta)_{i=1..\#\Delta}$; the first indicates a family of derivations of the parenthesized judgment indexed by the ith constructor of Δ and its constructor argument telescope $\overline{a_i : A_i}$, and the second merely names these telescopes explicitly and exhaustively. Γ^{G} indicates a typing context consisting of the definitions in Figures 8 and 9. *Italics* indicates meta-variables, `teletype` font indicates code literals (except in meta-variables denoting generated names like `Is/`D), and $^{\text{superscript}}$ denotes labels for meta-variables.

We use the following naming convention for expressions elaborated from datatypes and their constructors: $^{\mathrm{F}}$ for the usual impredicative encoding of a signature functor; $^{\mathrm{FI}}$ for the intersected version supporting functor induction; and $^{\mathrm{FIX}}$ for the least fixpoint of the inductive functor.

(a) $\boxed{\Gamma \vdash K \hookrightarrow K'}$ Kind elaboration

$$\frac{}{\Gamma \vdash \star \hookrightarrow \star} \qquad \frac{\Gamma \vdash K_1 \hookrightarrow K_1' \quad \Gamma, X{:}K_1 \vdash K_2 \hookrightarrow K_2'}{\Gamma \vdash \Pi\, X{:}K_1.\, K_2 \hookrightarrow \Pi\, X{:}K_1'.\, K_2'} \qquad \frac{\Gamma \vdash T : K_2 \hookrightarrow T' \quad \Gamma, x{:}T \vdash K_1 \hookrightarrow K_1'}{\Gamma \vdash \Pi\, x{:}T.\, K_1 \hookrightarrow \Pi\, x{:}T'.\, K_1'}$$

(b) $\boxed{\Gamma \vdash T : K \hookrightarrow T'}$ Type Elaboration (without datatypes)

$$\frac{FV(p_1\ p_2) \subseteq dom(\Gamma) \quad \Gamma \vdash p_1 \hookrightarrow p_1' \quad \Gamma \vdash p_2 \hookrightarrow p_2'}{\Gamma \vdash \{p_1 \simeq p_2\} : \star \hookrightarrow \{p_1' \simeq p_2'\}} \qquad \frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star \hookrightarrow T_2'}{\Gamma \vdash \iota\, x{:}T_1.\, T_2 : \star \hookrightarrow \iota\, x{:}T_1'.\, T_2'}$$

$$\frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star T_2'}{\Gamma \vdash \forall\, x{:}T_1.\, T_2 : \star \hookrightarrow \forall\, x{:}T_1'.\, T_2'} \qquad \frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star \hookrightarrow T_2'}{\Gamma \vdash \Pi\, x{:}T_1.\, T_2 : \star \hookrightarrow \Pi\, x{:}T_1'.\, T_2'}$$

$$\frac{\Gamma \vdash K \hookrightarrow K' \quad \Gamma, X : K \vdash T : \star \hookrightarrow T'}{\Gamma \vdash \forall\, X{:}K.\, T : \star \hookrightarrow \forall\, X{:}K'.\, T'} \qquad \frac{\Gamma \vdash S : K_1 \hookrightarrow S' \quad \Gamma, x{:}S \vdash T : K_2 \hookrightarrow T'}{\Gamma \vdash \lambda\, x{:}S.\, T : \Pi\, x{:}S.\, K_2 \hookrightarrow \lambda\, x{:}S'.\, T'}$$

$$\frac{\Gamma \vdash K_1 \hookrightarrow K_1' \quad \Gamma, X{:}K_1 \vdash T : K_2 \hookrightarrow T'}{\Gamma \vdash \lambda\, X{:}K_1.\, T : \Pi\, X{:}K_1.\, K_2 \hookrightarrow \lambda\, X{:}K_1'.\, T'} \qquad \frac{\Gamma \vdash T_1 : \Pi\, X{:}K_2.\, K_1 \hookrightarrow T_1' \quad \Gamma \vdash T_2 : K_2 \hookrightarrow T_2'}{\Gamma \vdash T_1 \cdot T_2 : [T_2/X]K_1 \hookrightarrow T_1' \cdot T_2'}$$

$$\frac{\Gamma \vdash T : \Pi\, x{:}S.\, K \hookrightarrow T' \quad \Gamma \vdash t : S \hookrightarrow t'}{\Gamma \vdash T\ t : [t/x]K \hookrightarrow T'\ t'} \qquad \frac{}{\Gamma \vdash X : \Gamma(X) \hookrightarrow X}$$

(a) $\boxed{\vdash \Gamma \hookrightarrow \Gamma'}$ Elaboration of contexts

$$\frac{}{\vdash \emptyset \hookrightarrow \emptyset} \qquad \frac{\vdash \Gamma \hookrightarrow \Gamma' \quad \Gamma \vdash T : \star \hookrightarrow T'}{\vdash \Gamma, x{:}T \hookrightarrow \Gamma', x{:}T'}$$

$$\frac{\vdash \Gamma \hookrightarrow \Gamma' \quad \Gamma \vdash K \hookrightarrow K'}{\vdash \Gamma, X{:}K \hookrightarrow \Gamma', X{:}K'} \qquad \frac{\vdash \Gamma \hookrightarrow \Gamma'}{\vdash \Gamma, \mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \hookrightarrow \Gamma', \mathcal{E}}$$

(b) $\boxed{\Gamma\,;\, (\overline{a{:}A}) \vdash \overline{s} : (\overline{a{:}B}) \hookrightarrow \overline{s'}}$ Elaboration of type-coerced constructor argument telescope

$$\frac{}{\Gamma\,;\, \varnothing \vdash \varnothing : \varnothing \hookrightarrow \varnothing} \qquad \frac{\Gamma, x{:}S \vdash s : T \hookrightarrow s' \quad \Gamma, x{:}S\,;\, (\overline{a{:}A}) \vdash \overline{s} : ([s/x]\overline{a{:}B}) \hookrightarrow \overline{s'}}{\Gamma\,;\, (x{:}S, \overline{a{:}A}) \vdash s, \overline{s} : (x{:}T, \overline{a{:}B}) \hookrightarrow s', \overline{s'}}$$

$$\frac{\Gamma, X{:}K_1 \vdash S : K_2 \hookrightarrow S' \quad \Gamma, X{:}K_1\,;\, (\overline{a{:}A}) \vdash \overline{s} : ([S/X]\overline{a{:}B}) \hookrightarrow \overline{s'}}{\Gamma\,;\, (X{:}K_1, \overline{a{:}A}) \vdash S, \overline{s} : (X{:}K_2, \overline{a{:}B}) \hookrightarrow S', \overline{s'}}$$

$$\frac{}{\Gamma \vdash x : \Gamma(x) \hookrightarrow x}$$

$$\frac{\Gamma \vdash S : \star \hookrightarrow S' \quad \Gamma, x{:}S \vdash t : T \hookrightarrow t'}{\Gamma \vdash \lambda\, x.\, t : \Pi\, x{:}S.\, T \hookrightarrow \lambda\, x.\, t'}$$

$$\frac{\Gamma \vdash S : \star \hookrightarrow S' \quad x \notin FV(|t|) \quad \Gamma, x{:}S \vdash t : T \hookrightarrow t'}{\Gamma \vdash \Lambda\, x.\, t : \forall\, x{:}S.\, T \hookrightarrow \Lambda\, x.\, t'}$$

$$\frac{\Gamma \vdash K \hookrightarrow K' \quad X \notin FV(|t|) \quad \Gamma, X{:}K \vdash t : T \hookrightarrow t'}{\Gamma \vdash \Lambda\, X.\, t : \forall\, X{:}K.\, T \hookrightarrow \Lambda\, X.\, t'}$$

$$\frac{\Gamma \vdash t : \Pi\, x{:}S.\, T \hookrightarrow t' \quad \Gamma \vdash s : S \hookrightarrow s'}{\Gamma \vdash t\ s : [s/x]T \hookrightarrow t'\ s'}$$

$$\frac{\Gamma \vdash t : \forall\, X{:}K.\, T \hookrightarrow t' \quad \Gamma \vdash S : K \hookrightarrow S'}{\Gamma \vdash t \cdot S : [S/X]T \hookrightarrow t' \cdot S'}$$

$$\frac{\Gamma \vdash t : \forall\, x{:}S.\, T \hookrightarrow t' \quad \Gamma \vdash s : S \hookrightarrow s'}{\Gamma \vdash t\ \text{-}s : [s/x]T \hookrightarrow t'\ \text{-}s'}$$

$$\frac{\Gamma \vdash t : S \hookrightarrow t' \quad S \cong T \quad \Gamma \vdash T : \star \hookrightarrow T'}{\Gamma \vdash t : T \hookrightarrow t'}$$

$$\frac{\Gamma \vdash t_1 : T_1 \hookrightarrow t_1' \quad \Gamma \vdash t_2 : [t_1/x]T_2 \hookrightarrow t_2' \quad t_1' \cong t_2'}{\Gamma \vdash [t_1, t_2] : \iota\, x{:}T_1.\, T_2 \hookrightarrow [t_1', t_2']}$$

$$\frac{\Gamma \vdash t : \iota\, x{:}T_1.\, T_2}{\Gamma \vdash t.1 : T_1 \hookrightarrow t'.1}$$

$$\frac{\Gamma \vdash t : \iota\, x{:}T_1.\, T_2 \hookrightarrow t'}{\Gamma \vdash t.2 : [t.1/x]T_2 \hookrightarrow t'.2}$$

$$\frac{}{\Gamma \vdash \{|t| \simeq |t|\} : \star \hookrightarrow \{p \simeq p\}}$$

$$\frac{}{\Gamma \vdash \beta : \{|t| \simeq |t|\} \hookrightarrow \beta}$$

$$\frac{\Gamma \vdash s : \{|t_1| \simeq |t_2|\} \hookrightarrow s' \qquad \Gamma \vdash t : [t_1/x]T_1 \hookrightarrow t' \qquad T_1 \cong T_2 \qquad \Gamma \vdash [t_2/x]T_2 : \star \hookrightarrow [t_2'/x]T_2'}{\Gamma \vdash \rho\ s\ @\ x.T_2\ \text{-}\ t : [t_2/x]T_2 \hookrightarrow \rho\ s'\ @\ x.T_2'\ \text{-}\ t'}$$

$$\frac{\Gamma \vdash s : \{|t_1| \simeq |t_2|\} \hookrightarrow s' \quad \Gamma \vdash t_1 : T \hookrightarrow t_1' \quad \Gamma \vdash t_2 \hookrightarrow t_2'}{\Gamma \vdash \varphi\ s - t_1\ \{t_2\} : T \hookrightarrow \varphi\ s' - t_1'\ \{t_2'\}}$$

$$\frac{\Gamma \vdash t : \{\lambda\, x.\, \lambda\, y.\, x \simeq \lambda\, x.\, \lambda\, y.\, x\} \hookrightarrow t' \quad \Gamma \vdash T : \star \hookrightarrow T'}{\Gamma \vdash \delta\ T - t : T \hookrightarrow \delta\ T' - t'}$$

Figure 13: $\boxed{\Gamma \vdash t : T \hookrightarrow t'}$ Term elaboration (without datatypes)

$$\frac{(c_i : \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\ D \in \Delta)_{i=1..\#\Delta} \quad (\Gamma, R : \star, X : \star \vdash \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\ X : \star \hookrightarrow \overset{\Pi}{\forall}\ \overline{a_i : A'_i}.\ X)_{i=1..\#\Delta}}{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{F}}{\hookrightarrow} \lambda\, R.\, \forall\, X : \star.\, (\Pi\, x_i : \overset{\Pi}{\forall}\ \overline{a_i : A'_i}.\ X)_{i=1..\#\Delta}.\ X}\ [\mathrm{F}]$$

$$\frac{c_j : \overset{\Pi}{\forall}\ \overline{a_j : A_j}.\ D \in \Delta}{\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], j) \overset{\mathrm{cF}}{\hookrightarrow} \Lambda\, R.\, \overset{\lambda}{\Lambda}\ \overline{a_j}.\ \Lambda\, X.\, \lambda\, x_{i=1..\#\Delta}.\, x_j\ \overline{a_j}}\ [\mathrm{cF}]$$

$$\frac{\begin{array}{c}\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{F}}{\hookrightarrow} D^{\mathrm{F}} \qquad (\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], i) \overset{\mathrm{cF}}{\hookrightarrow}\ c_i^{\mathrm{F}})_{i=1..\#\Delta} \\ (c_i : \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\ D \in \Delta)_{i=1..\#\Delta} \quad (\Gamma, R : \star, X : \star \vdash \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\ X : \star \hookrightarrow \overset{\Pi}{\forall}\ \overline{a_i : A'_i}.\ X)_{i=1..\#\Delta}\end{array}}{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FI}}{\hookrightarrow} \lambda\, R.\, \iota\, x : D^{\mathrm{F}} \cdot R.\, \forall\, X : D^{\mathrm{F}} \cdot R \to \star.\, (\Pi\, x_i : \overset{\Pi}{\forall}\ \overline{a_i : A'_i}.\ X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\ X\ x}\ [\mathrm{FI}]$$

$$\frac{\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], j) \overset{\mathrm{cF}}{\hookrightarrow} c_j^{F} \quad c_j : \overset{\Pi}{\forall}\ \overline{a_j : A_j}.\ D \in \Delta}{\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], j) \overset{\mathrm{cFI}}{\hookrightarrow} \Lambda\, R.\, \overset{\lambda}{\Lambda}\ \overline{a_j}.\, [c_j^{\mathrm{F}} \cdot R\ \overline{a_j}, \Lambda\, X.\, \lambda\, x_{i=1..\#\Delta}.\, x_j\ \overline{a_j}]}\ [\mathrm{cFI}]$$

$$\frac{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FI}}{\hookrightarrow}\ D^{\mathrm{FI}} \quad \Gamma \vdash D^{\mathrm{FI}} \overset{+}{\hookrightarrow} pos}{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow}\ \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos}\ [\mathrm{FIX}]$$

$$\frac{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow} \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos \quad \Gamma \vdash (\mathrm{Ind}[D, R, \Delta], j) \overset{\mathrm{cFI}}{\hookrightarrow}\ c_j^{\mathrm{FI}} \quad c_j : \overset{\Pi}{\forall}\ \overline{a_j : A_j}.\ D \in \Delta}{\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], j) \overset{\mathrm{cFIX}}{\hookrightarrow}\ \overset{\lambda}{\Lambda}\ \overline{a_j}.\ \ \mathtt{in} \cdot D^{\mathrm{FI}}\ \text{-}pos\ (c_j^{\mathrm{FI}} \cdot (\mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos)\ \overline{a_j})}\ [\mathrm{cFIX}]$$

$$\frac{\begin{array}{c}\Gamma \vdash \mathrm{Ind}[D, R, \Delta]\ wf \quad \Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow} \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos \quad (\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], i) \overset{\mathrm{cFIX}}{\hookrightarrow} c_i^{\mathrm{FIX}})_{i=1..\#\Delta} \\ \Theta = (\mathtt{Is}/D : \star \to \star,\ \mathtt{is}/D : \mathtt{Is}/D \cdot D,\ \mathtt{to}/D = \lambda\, x.\, x : \forall\, R : \star.\, \forall\, is : \mathtt{Is}/D \cdot R.\, R \to D) \\ \mathcal{E} = \left\{ \begin{array}{c} D \mapsto \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos, \quad (c_i \mapsto c_i^{\mathrm{FIX}})_{i=1..\#\Delta}, \\ \mathtt{Is}/D \mapsto \mathtt{IsD} \cdot D^{\mathrm{FI}}\ pos, \quad \mathtt{is}/D \mapsto \mathtt{isD} \cdot D^{\mathrm{FI}}\ \text{-}pos, \quad \mathtt{to}/D \mapsto \mathtt{toD} \cdot D^{\mathrm{FI}}\ \text{-}pos \end{array} \right\}\end{array}}{\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \dashv \Gamma, \mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}]}\ [\mathrm{Data}]$$

Figure 14: Elaboration of datatype declarations

(a) $\boxed{\Gamma \vdash F \overset{+}{\hookrightarrow} pos}$ Positivity

$$\frac{\Gamma, R{:}\star \vdash T : \star \hookrightarrow \_ \quad \Gamma, R_1{:}\star, R_2{:}\star, z{:}\mathtt{Cast}\cdot R_1 \cdot R_2 \ ; \ \mathtt{elimCast}\ \text{-}z \vdash [R_1/R]T \leqslant [R_2/R]T \hookrightarrow s'}{\Gamma \vdash \lambda\, R{:}\star.\, T \overset{+}{\hookrightarrow} \mathtt{intrMono}\ (\Lambda\, R_1.\, \Lambda\, R_2.\, \lambda\, z.\, \mathtt{intrCast}\ s'\ (\lambda\, \_.\, \beta))}$$

(b) $\boxed{\Gamma; s \vdash S \leqslant T \hookrightarrow s'}$ Subtyping

$$\frac{\Gamma \vdash T_1 \cong T_2 \quad \Gamma \vdash T_1 : \star \hookrightarrow \_ \quad \Gamma \vdash T_2 : \star \hookrightarrow \_}{\Gamma; s \vdash T_1 \leqslant T_2 \hookrightarrow \ \lambda\, x.\, x} \quad \frac{\Gamma \vdash s : S \to T \hookrightarrow \_ \quad |s| \cong \lambda\, x.\, x}{\Gamma; s \vdash S \leqslant T \hookrightarrow s}$$

$$\frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y{:}S_2; s' \vdash [(s\ y)/x]T_1 \leqslant [y/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \Pi\, x{:}S_1.\, T_1 \leqslant \Pi\, x{:}S_2.\, T_2 \hookrightarrow \lambda\, f.\, \lambda\, x.\, [x/y]t\ (f\ (s\ x))}$$

$$\frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y{:}S_2; s' \vdash [(s\ y)/x]T_1 \leqslant [y/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \forall\, x{:}S_1.\, T_1 \leqslant \forall\, x{:}S_2.\, T_2 \hookrightarrow \lambda\, f.\, \Lambda\, x.\, [x/y]t\ (f\ \text{-}(s\ x))}$$

$$\frac{\Gamma; s' \vdash S_1 \leqslant S_2 \hookrightarrow s \quad \Gamma, y{:}S_1; s' \vdash [y/x]T_1 \leqslant [(s\ y)/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \iota\, x{:}S_1.\, T_1 \leqslant \iota\, x{:}S_2.\, T_2 \hookrightarrow \lambda\, u.\, [s\ u.1, [u.1/y]t\ u.2]}$$

$$\frac{\Gamma; s' \vdash K_2 \leqslant K_1 \hookrightarrow S \quad \Gamma, Y : K_2; s' \vdash [(S \cdot Y)/X]T_1 \leqslant [Y/X]T_2 \hookrightarrow s}{\Gamma; s' \vdash \forall\, X{:}K_1.\, T_1 \leqslant \forall\, X{:}K_2.\, T_2 \hookrightarrow \lambda\, f.\, \Lambda\, X.\, [X/Y]s\ (f \cdot (S \cdot X))}$$

(c) $\boxed{\Gamma; s \vdash K_1 \leqslant K_2 \hookrightarrow S}$ Subkinding

$$\frac{}{\Gamma; s \vdash \star \ \leqslant \ \star \hookrightarrow \lambda\, X.\, X} \quad \frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y : S_2; s' \vdash [(s\ y)/x]K_1 \leqslant [y/x]K_2 \hookrightarrow S}{\Gamma; s' \vdash \Pi\, x{:}S_1.\, K_1 \leqslant \Pi\, x{:}S_2.\, K_2 \hookrightarrow \lambda\, P.\, \lambda\, x.\, [x/y]S \cdot (P\ (s\ x))}$$

$$\frac{\Gamma; s \vdash K_1' \leqslant \ K_1 \hookrightarrow S_1 \quad \Gamma, Y : K_1'; s \vdash [(S_1 \cdot Y)/X]K_2 \ \leqslant \ [Y/X]K_2' \hookrightarrow S_2}{\Gamma; s \vdash \Pi\, X{:}K_1.\, K_2 \leqslant \ \Pi\, X{:}K_1'.\, K_2' \hookrightarrow \lambda\, P.\, \lambda\, X{:}[X/Y]S_2 \cdot (P \cdot (S_1 \cdot X)).}$$

(d) $\boxed{\Gamma; s' \vdash (\overline{a{:}A_1}) \leqslant (\overline{a{:}A_2}) \hookrightarrow \overline{s}}$ Telescope coercion

$$\frac{}{\Gamma; s' \vdash \emptyset \hookrightarrow \emptyset} \quad \frac{\Gamma; s' \vdash A \leqslant A' \hookrightarrow s \quad \Gamma, y{:}A; s' \vdash \overline{a{:}[y/x]A_1} \leqslant \overline{a{:}[(s\ y)/x]A_2} \hookrightarrow \overline{s}}{\Gamma; s' \vdash (x{:}A, \overline{a{:}A_1}) \leqslant (x{:}A', \overline{a{:}A_2}) \hookrightarrow (s\ x), \overline{[x/y]s}}$$

$$\frac{\Gamma; s' \vdash K \leqslant K' \hookrightarrow S \quad \Gamma, Y{:}K; s \vdash \overline{a{:}[Y/X]A_1} \leqslant \overline{a{:}[(S \cdot Y)/X]A_2} \hookrightarrow \overline{s}}{\Gamma; s' \vdash (X{:}K, \overline{a{:}A_1}) \leqslant (X{:}K', \overline{x{:}A_2}) \hookrightarrow (S \cdot X), \overline{[X/Y]s}}$$

Figure 15: Positivity checker

(a) Lifting of properties of D^{FIX} to D^{F}

```
import DataInterface ·D^FI -pos.
Lift_D : Π P: D^FIX → ⋆. Π R: ⋆. Π is: IsD ·R. D^F ·R → ⋆
= λ P: D^FIX → ⋆. λ R: ⋆. λ is: IsD ·R. λ x: D^F ·R.
  ∀ m: D^FI ·R. ∀ eq: {m ≃ x}.
  P (in (φ eq - (toFD -is m) {x})).
```

(b) Elaboration of terms (μ' and μ)

$$\frac{\begin{array}{c}\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma, \mathcal{E}(D) = \mathtt{Fix}\cdot D^{\mathrm{FI}}\ pos, \mathtt{to/}D \in \Theta \quad \Gamma \vdash P : D \to \star \hookrightarrow P' \quad \Gamma \vdash t : D \hookrightarrow t' \\ \Gamma' = \Gamma, \mathtt{Type/}ih : \star, \mathtt{isType/}ih : \mathtt{Is/}D \cdot \mathtt{Type/}ih, ih : \Pi\ y{:}\mathtt{Type/}ih.\ P\ (\mathtt{to/}D\ \text{-}\mathtt{isType/}ih\ y) \\ \Gamma' \vdash \{c_i\ \overline{a_i} \to t_i\} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}, \mathtt{isType/}ih) \hookrightarrow (\{\overset{\lambda}{\Lambda}\ \overline{a_i}.\ t_i'\}, \mathtt{isType/}ih)\end{array}}{\begin{array}{c}\Gamma \vdash \mu\ ih.\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : P\ t \hookrightarrow \mathtt{mu} \cdot D^{\mathrm{FI}}\ \text{-}pos\ t' \cdot P' \\ (\Lambda\, \mathtt{Type/}ih.\, \Lambda\, \mathtt{isType/}ih.\, \lambda\, ih.\, \lambda\, x.\, x.2 \cdot (\mathtt{Lift}_D \cdot P' \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih)\ (\overset{\lambda}{\Lambda}\ \overline{a_i}.\ \Lambda\, m.\, \Lambda\, eq.\, t_i')_{i=1..\#\Delta}\ \text{-}x\ \text{-}\beta)\end{array}}\ [\textsc{Mu}]$$

$$\frac{\begin{array}{c}\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma,\ \mathcal{E}(D) = \mathtt{Fix}\cdot D^{\mathrm{FI}}\ pos, \mathtt{to/}D \in \Theta \quad \Gamma \vdash t : T \hookrightarrow t' \quad \Gamma \vdash T : \star \hookrightarrow T' \quad \Gamma \vdash P : D \to \star \hookrightarrow P' \\ \Gamma \vdash \{c_i\ \overline{a_i} \to t_i\} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}, is) \hookrightarrow (\{\overset{\lambda}{\Lambda}\ \overline{a_i}.\ t_i'\}, is')\end{array}}{\begin{array}{c}\mu'\texttt{<}is\texttt{>}\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : P\ (\mathtt{to/}D\ \text{-}is\ t) \hookrightarrow \mathtt{mu'} \cdot D^{\mathrm{FI}}\ \text{-}pos\ \text{-}is'\ t' \cdot P' \\ (\lambda\, x.\ x.2 \cdot (\mathtt{Lift}_D \cdot P'\ \cdot T'\ is')\ (\overset{\lambda}{\Lambda}\ \overline{a_i}.\ \Lambda\, m.\, \Lambda\, eq.\ t_i')_{i=1..\#\Delta}\ \text{-}x\ \text{-}\beta)\end{array}}\ [\textsc{Mu'}]$$

(c) Elaboration of datatypes, constructors, and globals in programs

$$\frac{\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma}{\Gamma \vdash D : \star \hookrightarrow \mathcal{E}(D)}\ [\mathrm{D}] \qquad \frac{\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma}{\Gamma \vdash \mathtt{Is/}D : \star \to \star \hookrightarrow \mathcal{E}(\mathtt{Is/}D)}\ [\mathrm{Is}] \qquad \frac{\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma}{\Gamma \vdash \mathtt{is/}D : \mathtt{Is/}D \cdot D \hookrightarrow \mathcal{E}(\mathtt{is/}D)}\ [\mathrm{is}]$$

$$\frac{\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma \quad c : \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\ D \in \Delta}{\Gamma \vdash c : \overset{\Pi}{\forall}\ \overline{a_i : [D/R]A_i}.\ D \hookrightarrow \mathcal{E}(c)}\ [\mathrm{C}] \qquad \frac{\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma}{\Gamma \vdash \mathtt{to/}D : \forall\, R{:}\star.\, \forall\, is{:}\mathtt{Is/}D \cdot R.\, R \to D \hookrightarrow \mathcal{E}(\mathtt{to/}D)}\ [\mathrm{to}]$$

Figure 16: $\boxed{\Gamma \vdash t : T \hookrightarrow t'}$ $\boxed{\Gamma \vdash T : K \hookrightarrow T'}$ Elaboration of datatype expressions

$$\frac{\Gamma \vdash t \hookrightarrow t' \quad (\Gamma \vdash t_i \hookrightarrow t'_i)_{i=1..n}}{\Gamma \vdash \mu\ ih.\ t\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n} \hookrightarrow |\texttt{mu}|\ t'(\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})}$$

$$\frac{\Gamma \vdash t \hookrightarrow t' \quad (\Gamma \vdash t_i \hookrightarrow t'_i)_{i=1..n}}{\Gamma \vdash \mu'\ t\ \{c_i\ \overline{a_i} \to t_i\}_{i=1..n} \hookrightarrow |\texttt{mu'}|\ t'\ (\lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})}$$

$$\frac{\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma}{\Gamma \vdash \texttt{is/}D \hookrightarrow |\mathcal{E}(\texttt{is/}D)|}\ [\mathrm{is}] \quad \frac{\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma \quad c : \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\, D \in \Delta}{\Gamma \vdash c \hookrightarrow |\mathcal{E}(c)|}\ [\mathrm{C}]$$

$$\frac{\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma}{\Gamma \vdash \texttt{to/}D \hookrightarrow |\mathcal{E}(\texttt{to/}D)|}\ [\mathrm{to}]$$

$$\frac{\Gamma, x : \{\lambda\, x.\, x \simeq \lambda\, x.\, x\} \vdash t \hookrightarrow p'}{\Gamma \vdash \lambda\, x.\, t \hookrightarrow \lambda\, x.\, p} \quad \frac{x : \_ \in \Gamma}{\Gamma \vdash x \hookrightarrow x} \quad \frac{\Gamma \vdash t_1 \hookrightarrow p_1 \quad \Gamma \vdash t_2 \hookrightarrow p_2}{\Gamma \vdash t_1\ t_2 \hookrightarrow p_1\ p_2}$$

Figure 17: $\boxed{\Gamma \vdash t \hookrightarrow p}$ Elaboration of untyped (post-erasure) expressions

5 Datatype Declarations

5.1 Datatype and Constructor Elaboration

Theorem 5.1 (Soundness: Elaboration of declarations)**.** *Assuming*

- $\Gamma \vdash \mathit{Ind}[D, R, \Delta] \dashv \Gamma, \mathit{IndEl}[D, R, \Delta, \Theta, \mathcal{E}]$, $(c_i : \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, D \in \Delta)_{i=1..\#\Delta}$
  (we have elaborated a well-formed datatype with constructors of a certain shape)
- $\vdash \Gamma \hookrightarrow \Gamma'$ *(the typing context elaborates, Figure 12a)*
- $(\Gamma, X{:}\star, R{:}\star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, X : \star \hookrightarrow \overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X$ *implies* $\Gamma^G, \Gamma', R{:}\star, X : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X : \star)_{i=1..\#\Delta}$
  (the elaborated constructor types are well-kinded)

we have that

- $\Gamma^G, \Gamma' \vdash \mathcal{E}(D) : \star$ *and* $(\Gamma^G, \Gamma' \vdash \mathcal{E}(c_i) : \overset{\Pi}{\forall}\, \overline{a_i : [\mathcal{E}(D)/R]A_i'}.\, \mathcal{E}(D))_{i=1..\#\Delta}$
  (the elaborated datatype and constructors have the expected kind and type)
- $\Gamma^G, \Gamma' \vdash \mathcal{E}(\texttt{Is/}D) : \star \to \star$, $\Gamma^G, \Gamma' \vdash \mathcal{E}(\texttt{is/}D) : \mathcal{E}(\texttt{Is/}D) \cdot \mathcal{E}(D)$
  (the elaborations of `Is/`D *and* `is/`D *have their expected kinds and types)*
- $\Gamma^G, \Gamma' \vdash \mathcal{E}(\texttt{to/}D) : \forall\, R{:}\star.\, \forall\, \mathit{is}{:}\mathcal{E}(\texttt{Is/}D) \cdot R.\, R \to \mathcal{E}(D)$, *with* $|\mathcal{E}(\texttt{to/}D)| =_{\beta\eta} \lambda\, x.\, x$
  (the elaboration of `to/`D *has its expected type and convertibility)*

Proof. The proof proceeds by traversing the derivation of the premises of rule [Data] and by mutual induction on the soundness of elaboration of terms and types (Theorem 6.2). We will write [*Rule*]*premise : classifier* to help guide the reader, and author(s).

Throughout the proof $\overline{a_i : A_i'}$ will refer to the elaboration of the construct c_i's argument telescope $\overline{a_i : A_i}$. We will also abbreviate $\texttt{Fix} \cdot D^{\mathrm{FI}}$ *pos* to D^{FIX}

[Data] $\Gamma \vdash \mathbf{Ind}[D, R, \Delta] \overset{\mathbf{FIX}}{\hookrightarrow} \texttt{Fix} \cdot D^{\mathbf{FI}}\ \mathit{pos} : \star$
The rule for deriving this premise is given by [FIX]. There, it suffices to show of the premises that

- [FIX] $\Gamma \vdash D^{\mathrm{FI}} : \star \to \star$
- [FIX] $\Gamma \vdash \texttt{intrMono}\ (\Lambda\, R_1\ R_2.\, \lambda\, z.\, \texttt{intrCast}\ t\ \lambda\, \_.\, \beta) : \texttt{Mono} \cdot D^{\mathrm{FI}}$

The second (which comes from inversion of the judgment form $\Gamma \vdash D^{\mathrm{FI}} \overset{+}{\hookrightarrow} \mathit{pos}$) is provable directly from Theorem 5.3, assuming the first; the first is proven below. Then $\texttt{Fix} \cdot D^{\mathrm{FI}}\ \texttt{intrMono}\ (\Lambda\, R_1\ R_2.\, \lambda\, z.\, \texttt{intrCast}\ t\ \lambda\, \_.\, \beta)$ has kind $\star$ as required.

[FIX] $\Gamma \vdash D^{\mathbf{FI}} : \star \to \star$
The rule for deriving this premise is given by [FI]. There, it suffices to show that

- [FI] $\Gamma \vdash D^{\mathrm{F}} : \star \to \star$
  Shown later in the proof
- [FI] $(\Gamma \vdash c_i^{\mathrm{F}} : \forall\, R{:}\star.\, \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, D^{\mathrm{F}} \cdot R)_{i=1..\#\Delta}$
  Shown later in the proof

To ensure that $D^{\mathrm{FI}} = \lambda\, R.\, \iota\, x{:}D^{\mathrm{F}} \cdot R.\, \forall\, X{:}D^{\mathrm{F}} \cdot R \to \star.\, \Pi\, x_i{:}(\overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, X\ x$
has kind $\star \to \star$, show the two components of the intersection have kind $\star$ when $R{:}\star$. $D^{\mathrm{F}} \cdot R$ does by

- [FI] $\Gamma \vdash D^{\mathrm{F}} : \star \to \star$

The rhs $\forall\, X : D^{\mathrm{F}} \cdot R \to \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, X\ x$ obviously has kind $\star$ under a context extended by $x : D^{\mathrm{F}} \cdot R$ (by combining premise of [FI] $(\Gamma, R : \star, X : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, X : \star \hookrightarrow \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X)_{i=1..\#\Delta}$ with the assumption that this means that $(\Gamma^{\mathrm{G}}, \Gamma', R : \star, X : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X : \star)_{i=1..\#\Delta}$, and by an easy inductive argument that this means $\Gamma^{\mathrm{G}}, \Gamma', R : \star, X : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}) : \star)_{i=1..\#\Delta}$ also) if we can show the elaborated constructors c_i^{F} are well-typed. By

- [FI] $(\Gamma \vdash c_i^{\mathrm{F}} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathrm{F}} \cdot R)_{i=1..\#\Delta}$ (shown further below)

the application $c_i^{\mathrm{F}}\ \overline{a_i}$ in the codomain of the abstract constructor has type $D^{\mathrm{F}} \cdot R$

Thus the elaborated type has kind $\star \to \star$

[FI] $\Gamma \vdash D^{\mathbf{F}} : \star \to \star$

The rule for deriving this premise is given by [F].

It is clear that $D^{\mathrm{F}} = \lambda\, R.\, \forall\, X : \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X)_{i=1..\#\Delta}.\, X$ has the desired kind $\star \to \star$, as we assumed $(\Gamma, X : \star, R : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, X : \star \hookrightarrow \overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X$ (which we have as a premise of [FI]) implies $\Gamma^{\mathrm{G}}, \Gamma', R : \star, X : \star \vdash \overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X : \star)_{i=1..\#\Delta}$ at the start of the proof.

[FI] $(\Gamma \vdash c_i^{\mathbf{F}} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathbf{F}} \cdot R)_{i=1..\#\Delta}$

The rule for deriving these premises is given by [cF]. There, we will consider each constructor $c_j \in \Delta$.

To conclude, we want term $\Lambda\, X.\, \lambda\, \overline{x_i}.\, x_j\ \overline{a_j}$ to have type $D^{\mathrm{F}} \cdot R$ under a context extended by $R : \star, \overline{a_j : A'_j}$. Notice that, from

- [FI] $\Gamma \vdash D^{\mathrm{F}} : \star \to \star$

above we have that $D^{\mathrm{F}} \cdot R = \forall\, X : \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X)_{i=1..\#\Delta}.\, X$ has kind $\star$. Now we further extend the context by $X : \star, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X)_{i=1..\#\Delta}$ and show $x_j\ \overline{a_j}$ has type X – which it does

[Data] $(\Gamma \vdash c_i^{\mathbf{FIX}} : \overset{\Pi}{\forall}\, \overline{a_i : [D^{\mathbf{FIX}}/R]A'_i}.\, D^{\mathbf{FIX}})_{i=1..\#\Delta}$

We are now at the second premise of rule [Data]. The rule for deriving this premise is [cFIX]. There it suffices to show the following for the premises:

- [cFIX] $\Gamma \vdash D^{\mathrm{FIX}} : \star$

  Satisfied by [Data] $\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow} D^{\mathrm{FIX}} : \star$ above

- [cFIX] $\Gamma \vdash c_j^{\mathrm{FI}} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathrm{FI}} \cdot R$

  Shown later in the proof

We must show that under a context extended by $\overline{a_j : [D^{\mathrm{FIX}}/R]A'_j}$, the expression $\mathtt{in} \cdot D^{\mathrm{FI}}\text{-}pos\ (c_j^{\mathrm{FI}}\ \overline{a_j})$ has type $D^{\mathrm{FIX}} = \mathtt{Fix} \cdot D^{\mathrm{FI}}\text{-}pos$

We know from the discussion of [Data] $\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow} \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos : \star$, meaning $D^{\mathrm{FI}} : \star \to \star$ and $pos : \mathtt{Mono} \cdot D^{\mathrm{FI}}$, so these first two arguments to $\mathtt{in}$ (in the definition of c_j^{FIX}, the elaboration of rule [cFIX]) are type-correct, and the remaining argument $c_j^{\mathrm{FI}}\ \overline{a_j}$ must have type $D^{\mathrm{FI}} \cdot D^{\mathrm{FIX}}$. This is follows by the type of c_j^{FI} above

[cFIX] $\Gamma \vdash\ c_j^{\mathbf{FI}} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathbf{FI}} \cdot R$

The rule for deriving these premises is given by [cFI]. There, it suffices to show uniformly for the j-th constructor the following for the premises

- [cFI] $\Gamma \vdash c_j^{F} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_j : A'_j}.\, D^{\mathrm{F}} \cdot R$

  Proved above in case [FI] $(\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], i) \overset{\mathrm{cF}}{\hookrightarrow}\ c_i^{\mathrm{F}} : \forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathrm{F}} \cdot R)_{i=1..\#\Delta}$

To show that the elaboration $c_j^{\mathrm{FI}}\ (\Lambda\, R.\, \overset{\lambda}{\Lambda}\, \overline{a_j}.\, [c_j^{\mathrm{F}}\ \overline{a_j}, \Lambda\, X.\, \lambda\, x_{i=1..\#\Delta}.\, x_j\ \overline{a_j}])$ has the expected type $\forall\, R : \star.\, \overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, D^{\mathrm{FI}} \cdot R$ (where $D^{\mathrm{FI}} = \lambda\, R.\, \iota\, x : D^{\mathrm{F}} \cdot R.\, \forall\, X : D^{\mathrm{F}} \cdot R \to \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A'_i}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, X\ x$)

Extend the context by $R : \star$ and see that the intersection introduction form has the expected intersection type.

- For the lhs of the intersection, we already know from above what the type and shape of c_j^{F} is.

  The the type of $c_j^{\mathrm{F}}\ \overline{a_j}$ is $D^{\mathrm{F}} \cdot R$, and it erases to $\lambda\, \overline{x_{i=1..\#\Delta}}.\, x_j\ \overline{|a_j|}$

- For the rhs of the intersection, it is immediate that it has the same erasure as the lhs, so we must determine whether it has the expected type $\forall\, X : D^{\mathrm{F}} \cdot R \to \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : A_i'}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, X\ (c_j^{\mathrm{F}}\ \overline{a_j})$

  Extending the context with X and $\overline{x_{i=1..\#\Delta}}$ of the appropriate type, it is clear that $x_j\ \overline{a_j}$ has the type $X\ (c^{\mathrm{F}}\ \overline{a_j})$

Thus the expression does have the expected type.

[Data] $\mathcal{E} = \{\cdots\}$ In the proofs for the first two premises of [Data], we saw

- $\Gamma \vdash \mathrm{Ind}[D, R, \Delta] \overset{\mathrm{FIX}}{\hookrightarrow} \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos : \star$

  Since by assumption $D : \star$, the elaborated type of D has the elaborated kind of D

- $(\Gamma \vdash (\mathrm{Ind}[D, R, \Delta], i) \overset{\mathrm{cFIX}}{\hookrightarrow} c_i^{\mathrm{FIX}} : \overset{\Pi}{\forall}\, \overline{a_i : [\mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos/R]A_i'}.\, \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos)_{i=1..\#\Delta}$

  Since each $c_i : \overset{\Pi}{\forall}\, \overline{a_i : A_i}.\, D \in \Delta$, it is clear that its elaboration c_i^{FIX} has the type of the elaboration of c_i's type where R is replaced by D.

- For $\mathtt{Is}/D : \star \to \star$, its elaboration $\mathtt{IsD} \cdot D^{\mathrm{FI}}\ pos$ has kind $\star \to \star$, as D^{FI} was shown already to have kind $\star \to \star$ and pos type $\mathtt{Mono} \cdot D^{\mathrm{FI}}$

- For $\mathtt{is}/D : \mathtt{Is}/D \cdot D$, it is clear its elaboration $\mathtt{isD} \cdot D^{\mathrm{FI}}\ pos$ has type $\mathtt{IsD} \cdot D^{\mathrm{FI}}\ pos \cdot (\mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos)$

- For $\mathtt{to}/D : \forall\, R : \star.\, \forall\, is : \mathtt{Is}/D \cdot R.\, R \to D$, it is clear that its elaboration $\mathtt{toD} \cdot D^{\mathrm{FI}}\ pos$ has type $\forall\, R : \star.\, \forall\, is : \mathtt{IsD} \cdot D^{\mathrm{FI}}\ pos.\, R \to \mathtt{Fix} \cdot D^{\mathrm{FI}}\ pos$,

  Furthermore, thanks to erasure its elaboration is convertible with $\lambda\, x.\, x$

□

Theorem 5.2. *(Soundness: Convertibility of constructors)*
If $\Gamma \vdash c_j \cong {c'}_k$ *and* $\Gamma \vdash c_j : \overset{\Pi}{\forall} \overline{a_j : A_j}.\, D \hookrightarrow c_j^{FIX}$ *and* $\Gamma \vdash {c'}_k : \overset{\Pi}{\forall} \overline{a_k : A'_k}.\, D' \hookrightarrow {c'}_k^{FIX}$ *then* $|c_j^{FIX}| =_{\beta\eta} |{c'}_k^{FIX}|$

Proof. We first note that, as both constructors are formed using the generic `in` of Figure 9 (and that two arguments $F : \star \to \star$ and $\{im : \texttt{Mono}\ \cdot F\}$ given as module parameters erase away), the proof reduces to showing that the the two intersection-functor constructors c_j^{FI} and ${c'}_k^{\mathrm{FI}}$ are convertible. These are elaborated as

- $\Lambda\, R.\, \overset{\lambda}{\Lambda}\, \overline{a_j}.\, [c^{\mathrm{F}} \cdot R\ \overline{a_j}, \Lambda\, X.\, \lambda\, x_{i=1..\#\Delta}..x_j\ \overline{a_j}]$
- $\Lambda\, R.\, \overset{\lambda}{\Lambda}\, \overline{a_k}.\, [c^{\mathrm{F}\prime} \cdot R\ \overline{a_k}, \Lambda\, X.\, \lambda\, x_{i=1..\#\Delta'}..x_k\ \overline{a_k}]$

By erasure of terms (and argument sequences) and convertibility of dependent intersection introduction, this reduces to showing these two terms are convertible

- $\lambda\, |\overline{a_j}|.\, \lambda\, x_{i=1..\#\Delta}..x_j\ |\overline{a_j}|$
- $\lambda\, |\overline{a_k}|.\, \lambda\, x_{i=1..\#\Delta'}..x_k\ |\overline{a_k}|$

By the assumed derivation of convertibility of constructors, $j = k$, $\#|\overline{a_k}| = \#|\overline{a_j}|$, and $\#\Delta = \#\Delta'$. Thus the two terms are convertible □

5.2 Positivity Checker

Theorem 5.3 (Soundness: Positivity checker)**.**

1. *If* $\Gamma; s \vdash S \leqslant T \hookrightarrow s'$ *then* $\Gamma \vdash s' : S \to T \hookrightarrow \_$ *and* $|s'| =_{\beta\eta} \lambda\, x.\, x$
2. *If* $\Gamma; s \vdash K_1 \leqslant K_2 \hookrightarrow S$ *then* $\Gamma \vdash S : K_1 \to K_2 \hookrightarrow \_$
3. *If* $\Gamma \vdash F \overset{+}{\hookrightarrow} pos$ *then* $\Gamma, \Gamma^G \vdash pos : \texttt{Mono} \cdot F \hookrightarrow \_$
4. [1] *If* $\Gamma; s' \vdash \overline{a : A} \leqslant \overline{a : A'} \hookrightarrow \overline{s}$ *then* $\Gamma; (\overline{a : A}) \vdash \overline{s} : (\overline{a : A'}) \hookrightarrow \_$ *and* $|\overline{s}| \cong |\overline{a}|$

Proof. By a straightforward mutual induction on the assumed derivation. A few example cases are given

Part 1. Case

$$\frac{\Gamma \vdash T_1 \cong T_2 \quad \Gamma \vdash T_1 : \star \hookrightarrow \_ \quad \Gamma \vdash T_2 : \star \hookrightarrow \_}{\Gamma; s \vdash T_1 \leqslant T_2 \hookrightarrow \ \lambda\, x.\, x}$$

Appeal to convertibility and function rules of Figure 13

Part 1. Case

$$\frac{\Gamma \vdash s : S \to T \hookrightarrow \_ \quad |s| \cong \lambda\, x.\, x}{\Gamma; s \vdash S \leqslant T \hookrightarrow s}$$

From the premises

Part 1. Case

$$\frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y : S_2; s' \vdash [(s\ y)/x]T_1 \leqslant [y/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \Pi\, x : S_1.\, T_1 \leqslant \Pi\, x : S_2.\, T_2 \hookrightarrow \lambda\, f.\, \lambda\, x.\, [x/y]t\ (f\ (s\ x))}$$

- By the IH, s has type $S_2 \to S_1$ and reducible with $\lambda\, x.\, x$
- By the IH, (under context extended by $y : S_2$), t has type $[(s\ y)/x]T_1 \to [y/x]T_2$ and reducible with $\lambda\, x.\, x$
- Assume f has type $\Pi\, x : S_1.\, T_1$ and x type S_2
- $f\ (s\ x)$ has type $[(s\ x)/x]T_1$ and reducible with $f\ x$
- $[x/y]t$ of this has type T_2, reducible to the same
- η-contract and conclude

Part 1. Case

$$\frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y : S_2; s' \vdash [(s\ y)/x]T_1 \leqslant [y/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \forall\, x : S_1.\, T_1 \leqslant \forall\, x : S_2.\, T_2 \hookrightarrow \lambda\, f.\, \Lambda\, x.\, [x/y]t\ (f\ \text{-}(s\ x))}$$

Similar to above

Part 1. Case

$$\frac{\Gamma; s' \vdash K_2 \leqslant K_1 \hookrightarrow S \quad \Gamma, Y : K_2; s' \vdash [(S \cdot Y)/X]T_1 \leqslant [Y/X]T_2 \hookrightarrow s}{\Gamma; s' \vdash \forall\, X : K_1.\, T_1 \leqslant \forall\, X : K_2.\, T_2 \hookrightarrow \lambda\, f.\, \Lambda\, X.\, [X/Y]s\ (f \cdot (S \cdot X))}$$

- By mutual induction on 2., S has kind $K_2 \to K_1$
- By IH, s (under context extended by Y of kind K_2) has type $[S \cdot Y/X]T_1 \to [Y/X]T_2$, and reducible with $\lambda\, x.\, x$
- Assume $f : \forall\, X : K_1.\, T_1$, $X : K_2$
- $f \cdot (S \cdot X)$ has kind $[S \cdot X/X]T_1$, reducible (after erasure!) with f
- $[X/Y]s$ of this has type T_2, reducible with the same
- Erase to get conversion with $\lambda\, f.\, f$

[1] c.f. Figure 12b

Part 1. Case

$$\frac{\Gamma; s' \vdash S_1 \leqslant S_2 \hookrightarrow s \quad \Gamma, y : S_1; s' \vdash [y/x]T_1 \leqslant [(s\ y)/x]T_2 \hookrightarrow t}{\Gamma; s' \vdash \iota\, x{:}S_1.\, T_1 \leqslant \iota\, x{:}S_2.\, T_2 \hookrightarrow \lambda\, u.\, [s\ u.1, [u.1/y]t\ u.2]}$$

By assumption $\Gamma \vdash s : S_1 \to S_2$ and $s =_{\beta\eta} \lambda\, x.\, x$, and $\Gamma, y{:}S_1 \vdash t : [y/x]T_1 \to [(s\ y)/x]T_2$ and $t =_{\beta\eta} \lambda\, x.\, x$.

Thus, the two components of the intersection in the body of the elaborated term have type S_2 and $[(s\ u.1)/x]T_2$ (and this type is convertible with $[u.1/x]T_2$), and are convertible with $u.1$ and $u.2$. Thus the entire expression is convertible with $\lambda\, u.\, u$ and can be assigned type $\iota\, x{:}S_1.\, T_1 \to \iota\, x{:}S_2.\, T_2$

Part 2. Case

$$\frac{}{\Gamma; s \vdash \star \ \leqslant\ \star \hookrightarrow \lambda\, X.\, X}$$

Immediate

Part 2. Case

$$\frac{\Gamma; s' \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y : S_2; s' \vdash [(s\ y)/x]K_1 \leqslant [y/x]K_2 \hookrightarrow S}{\Gamma; s' \vdash \Pi\, x{:}S_1.\, K_1 \leqslant \Pi\, x{:}S_2.\, K_2 \hookrightarrow \lambda\, P.\, \lambda\, x.\, [x/y]S \cdot (P\ (s\ x))}$$

- By mutual induction on 1., $s : S_2 \to S_1$ and convertible with $\lambda\, x.\, x$
- By IH, (under context extended by $y{:}S_2$), S has kind $[(s\ y)/x]K_1 \to [y/x]K_2$
- Assume $P : \Pi\, x{:}S_1.\, K_1$, $x : S_2$
- $P\ (s\ x)$ has kind $[s\ x/x]K_1$
- $[x/y]S$ of this has kind K_2

Part 2. Case

$$\frac{\Gamma; s \vdash K_1' \leqslant\ K_1 \hookrightarrow S_1 \quad \Gamma, Y : K_1'; s \vdash [(S_1 \cdot Y)/X]K_2 \ \leqslant\ [Y/X]K_2' \hookrightarrow S_2}{\Gamma; s \vdash \Pi\, X{:}K_1.\, K_2 \leqslant\ \Pi\, X{:}K_1'.\, K_2' \hookrightarrow \lambda\, P.\, \lambda\, X{:}[X/Y]S_2 \cdot (P \cdot (S_1 \cdot X)).}$$

By assumption $\Gamma \vdash S_1 : K_1' \to K_1$ and $S_1 =_{\beta\eta} \lambda\, X.\, X$,
and $\Gamma, Y : K_1' \vdash S_2 : [(S_1 \cdot Y)/X]K_2 \ \to [Y/X]K_2'$ and $S_2 =_{\beta\eta} \lambda\, X.\, X$. It is clear then that the elaborated type in the conclusion has kind $\Pi\, X{:}K_1.\, K_2 \to \Pi\, X{:}K_1'.\, K_2'$, and reduces and η-contracts to $\lambda\, P.\, P$

Part 3. Case

$$\frac{\Gamma, R{:}\star \vdash T : \star \hookrightarrow \_ \quad \Gamma, R_1{:}\star, R_2{:}\star, z{:}\mathtt{Cast} \cdot R_2 \cdot R_1\ ;\ \mathtt{elimCast}\ \text{-}z \vdash [R_1/R]T \leqslant [R_2/R]T \hookrightarrow t}{\Gamma \vdash \lambda\, R{:}\star.\, T \overset{+}{\hookrightarrow} \Lambda\, R_1\ R_2.\, \lambda\, z.\, \mathtt{intrCast}\ t\ \lambda\, \_.\, \beta}$$

We appeal to the proof of 1. (noting that the premises of the "base" rule will be satisfid as `elimCast` $\text{-}z$ has the required type and erasure). Then, t has type $F \cdot R_1 \to F \cdot R_2$, $\lambda\, \_.\, \beta$ is a proof that that $\Pi\, R_1{:}\star.\, \{t\ r \simeq r\}$, and thus the whole elaborated expression has type $\mathtt{Mono} \cdot F$, if we make sure that such definitions as `Cast`, `elimCast`, and `Mono` are in the context – thus the extension to context $\Gamma, \Gamma^{\mathrm{G}}$.

Part 4. Case

$$\frac{}{\Gamma; s' \vdash \emptyset \hookrightarrow \emptyset}$$

Trivial

Part 4. Case

$$\frac{\Gamma; s' \vdash A \leqslant A' \hookrightarrow s \quad \Gamma, y : A; s' \vdash \overline{a : [y/x]A_1} \leqslant \overline{a : [(s\ y)/x]A_2} \hookrightarrow \overline{s}}{\Gamma; s' \vdash x : A, \overline{a : A_1} \leqslant x : A', \overline{a : A_2} \hookrightarrow (s\ x), \overline{[x/y]s}}$$

So, these are pretty nasty. The difficulty is mostly notational, not theoretical – that is, describing the coercion, and typing, of telescope of constructor arguments.

- Appealing to 1., we have s has type $A \to A'$ and erases to $\lambda\, x.\, x$

  This means $(s\ x)$ has type A' and erases to x
- Appealing to the inductive hypothesis, we have $\Gamma, y : A\,;\, (\overline{a : A_1}) \vdash \overline{s} : (\overline{a : [s\ y/x]A_2}) \hookrightarrow \_$ (Figure 12b) and $\overline{s}$ converts with $\overline{a}$
- Conclude $\Gamma\,;\, (x : A, \overline{a : A_1}) \vdash (s\ x), \overline{s} : (x : A', \overline{a : A_2}) \hookrightarrow \_$

Part 4. Case

$$\frac{\Gamma, X : K_1 \vdash S : K_2 \hookrightarrow S' \quad \Gamma, X : K_1\,;\, (\overline{a : A}) \vdash \overline{s} : ([S/X]\overline{a : B}) \hookrightarrow \overline{s'}}{\Gamma\,;\, (X : K_1, \overline{a : A}) \vdash S, \overline{s} : (X : K_2, \overline{a : B}) \hookrightarrow S', \overline{s'}}$$

Similar to above □

5.3 Additional Proofs

Lemma 5.4. *(Reconstruction of coercions from datatype elaboration) Assuming some* Γ, Γ', *IndEl*$[D, R, \Delta, \Theta, \mathcal{E}]$, U, U' *and term "*is*" such that*

- $\vdash \Gamma \hookrightarrow \Gamma'$
- *IndEl*$[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma$, $\texttt{to/}D \in \Theta$, $\texttt{Is/}D \in \Theta$, $\mathcal{E}(D) = \texttt{Fix} \cdot D^{FI}$ *pos*
- $\Gamma \vdash U : \star \hookrightarrow U'$ *and* $\Gamma \vdash is : \texttt{Is/}D \cdot U \hookrightarrow is'$, *and* $\Gamma^G, \Gamma' \vdash U' : \star$

we have the following:

- *If* $\Gamma; \texttt{to/}D$ *-is* $\vdash S \leqslant T \hookrightarrow s$, *where* $\Gamma \vdash S : \star \hookrightarrow S'$, $\Gamma \vdash T : \star \hookrightarrow T'$, $\Gamma^G, \Gamma' \vdash S' : \star$, *and* $\Gamma^G, \Gamma' \vdash T' : \star$
  then there is some s' *such that* $\Gamma \vdash s : S \to T \hookrightarrow s'$ *and* $\Gamma^G, \Gamma'; \texttt{toD} \cdot D^{FI}$ *-pos -is'* $\vdash S' \leqslant T' \hookrightarrow s'$
- *If* $\Gamma; \texttt{to/}D$ *-is* $\vdash K_1 \leqslant K_2 \hookrightarrow S$, *where* $\Gamma \vdash K_1 \hookrightarrow K_1'$, $\Gamma \vdash K_2 \hookrightarrow K_2'$, $\Gamma^G, \Gamma' \vdash K_1'$, *and* $\Gamma^G, \Gamma' \vdash K_2'$
  then there is some S' *such that* $\Gamma \vdash S : K_1 \to K_2 \hookrightarrow S'$
  and $\Gamma^G, \Gamma'; \texttt{toD} \cdot D^{FI}$ *-pos -is'* $\vdash {K_1}' \leqslant {K_2}' \hookrightarrow S'$,

That is, that constructor-argument coercions produced from base assumption $\texttt{to/}D$ *for types and kinds in the surface language elaborate to coercions produced by base assumption* $\texttt{toD}$ *for elaborated their elaborations.*

Proof. By induction on the assumed derivation of the coercion. This lemma is used by Theorem 6.2. Some interesting cases are given below

$$\textbf{Case} \quad \frac{\Gamma \vdash \texttt{to/}D \text{ -}is : U \to D \quad |\texttt{to/}D \text{ -}is| = \lambda\, x.\, x}{\Gamma; \texttt{to/}D \text{ -}is \vdash U \leqslant D \hookrightarrow \texttt{to/}D \text{ -}is}$$

The elaboration of $\texttt{to/}D$ *-is* is $\texttt{toD} \cdot D^{\mathrm{FI}}$ *-pos -is'*, as it has the desired type and erasure under the elaborated context $\Gamma^{\mathrm{G}}, \Gamma'$ by assumption

$$\frac{\Gamma^{\mathrm{G}}, \Gamma' \vdash \texttt{toD} \cdot D^{\mathrm{FI}} \text{ -}pos \text{ -}is' : S' \to T' \quad |\texttt{toD} \cdot D^{\mathrm{FI}} \text{ -}pos \text{ -}is'| = \lambda\, x.\, x}{\Gamma^{\mathrm{G}}, \Gamma'; \texttt{toD} \cdot D^{\mathrm{FI}} \text{ -}pos \text{ -}is' \vdash S' \leqslant T' \hookrightarrow \texttt{toD} \cdot D^{\mathrm{FI}} \text{ -}pos \text{ -}is'}$$

$$\textbf{Case} \quad \frac{\Gamma; \texttt{to/}D \text{ -}is \vdash S_2 \leqslant S_1 \hookrightarrow s \quad \Gamma, y : S_2; \texttt{to/}D \text{ -}is \vdash [(s\ y)/x]T_1 \leqslant [y/x]T_2 \hookrightarrow t}{\Gamma; \texttt{to/}D \text{ -}is \vdash \Pi\, x : S_1.\, T_1 \leqslant \Pi\, x : S_2.\, T_2 \hookrightarrow \lambda\, f.\, \lambda\, y.\, t\ (f\ (s\ y))}$$

Our assumptions are:

1. $\vdash \Gamma \hookrightarrow \Gamma'$
2. $\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma, \texttt{to/}D \in \Theta, \texttt{Is/}D \in \Theta, \mathcal{E}(D) = \texttt{Fix} \cdot D^{\mathrm{FI}}$ *pos*
3. $\Gamma \vdash U : \star \hookrightarrow U'$ and $\Gamma \vdash is : \texttt{Is/}D \cdot U \hookrightarrow is'$, and $\Gamma^{\mathrm{G}}, \Gamma' \vdash U' : \star$
4. $\Gamma \vdash \Pi\, x : S_1.\, T_1 : \star \hookrightarrow \Pi\, x : S_1'.\, T_1'$ (inversion of elaboration for types)
5. $\Gamma \vdash \Pi\, x : S_2.\, T_2 : \star \hookrightarrow \Pi\, x : S_2'.\, T_2'$ (inversion of elaboration for types)
6. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, x : S_2'.\, T_2' : \star$
   With this and assumptions 1. and 4., we know $\vdash \Gamma, y : S_2 \hookrightarrow \Gamma', y : S_2'$ and $\Gamma^{\mathrm{G}}, \Gamma', y : S_2' \vdash [y/x]T_2' \vdash \star$
7. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, x : S_1'.\, T_1' : \star$
   With this and assumptions 1., 5., and 6., we know that $\Gamma^{\mathrm{G}}, \Gamma', y : S_2' \vdash [s'\ y/x]T_2' \vdash \star$ for any $s : S_2' \to S_1'$

Invoke the induction hypothesis on s and t to get their elaborations s' and t', noting that for the latter the elaborated context is $\vdash \Gamma, y : S_2 \hookrightarrow \Gamma', y : S_2'$ and that (from the IH) $s' : S_2' \to S_1'$ under the elaborated context. Then, the coercion we want is $\lambda\, f.\, \lambda\, y.\, t'\ (f\ (s'\ y))$, and this is obviously the elaboration of $\lambda\, f.\, \lambda\, y.\, t\ (f\ (s\ y))$

Case $$\frac{\Gamma; \texttt{to/}D\text{ -}is \vdash K_3 \leqslant\ K_1 \hookrightarrow S_1 \quad \Gamma, Y : K_3; \texttt{to/}D\text{ -}is \vdash [(S_1 \cdot Y)/X]K_2\ \leqslant\ [Y/X]K_4 \hookrightarrow S_2}{\Gamma; \texttt{to/}D\text{ -}is \vdash \Pi\, X : K_1.\, K_2 \leqslant\ \Pi\, X : K_3.\, K_4 \hookrightarrow \lambda\, P.\, \lambda\, X : [X/Y]S_2 \cdot (P \cdot (S_1 \cdot X)).}$$

This follows by a similar argument to the case above. Invoke the induction hypothesis on S_1 and S_2 to get their elaborations S_1' and T_2', noting that for the latter the elaborated context is $\Gamma', y : K_3' \vdash \Gamma, Y : K_3 \hookrightarrow \Gamma', y : K_3'$ (where we know K_3 is the elaboration of K_3 by assumption and by inversion of the elaboration rules). Then, the coercion we want is $\lambda\, P.\, \lambda\, X.\, [X/Y]S_2' \cdot (P \cdot (S_1' \cdot X))$, which is obviously the elaboration of $\lambda\, P.\, \lambda\, X : [X/Y]S_2 \cdot (P \cdot (S_1 \cdot X))$.

□

Corolary 5.4.1. *Lemma 5.4 also holds when extended to a sequence of coercions $\overline{s}$ for telescope $\overset{\Pi}{\forall}\ \overline{a : A_1}.\ \leqslant \overset{\Pi}{\forall}\ \overline{a : A_2}$.*

Proof. By a straighforward induction over the rules of Figure 12b, appealing to Lemma 5.4 for each non-empty telescope.

□

6 Functions over Dataypes

6.1 Value preservation

Theorem 6.1 (Soundness: Value-preservation of μ and μ'). *The elaborated λ-expresions of μ- and μ'-expressions are confluent with the elaborations of the terms they step to, i.e.:*

- *If* $\Gamma \vdash \mu\ ih.\ (c_j\ \overline{s})\ \{\ c_i\ \overline{a} \to t_i\}_{i=1..n} \hookrightarrow e_1$, *and* $\mu\ ih.\ (c_j\ \overline{s})\ \{\ c_i\ \overline{a} \to t_i\}_{i=1..n} \rightsquigarrow t$, *and* $\Gamma \vdash t \hookrightarrow e_2$, *then there exists some* e_3 *such that* $e_1 \rightsquigarrow^* e_3$ *and* $e_2 \rightsquigarrow^* e_3$
- *If* $\Gamma \vdash \mu'\ (c_j\ \overline{s})\ \{c_i\ \overline{a_i} \to t_i\}_{i=1..n} \hookrightarrow e_1$, *and* $\mu'\ (c_j\ \overline{s})\ \{c_i\ \overline{a_i} \to t_i\}_{i=1..n} \rightsquigarrow t$, *and* $\Gamma \vdash t \hookrightarrow e_2$, *then there exists some* e_3 *such that* $e_1 \rightsquigarrow^* e_3$ *and* $e_2 \rightsquigarrow^* e_3$

Proof.

Case μ: By elaboration of untyped μ-expressions in Figure 17, we know that our μ-expression elaborates

$$e_1 = |\texttt{mu}|\ (c_j^{\text{FIX}}\ \overline{s'})\ (\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n})$$

and by inversion of reduction rule for μ we have the form of the single step reduction of the μ-expressions is forced by the following derivation

$$\frac{1 \leq j \leq n \quad \#\overline{s} = \#\overline{a_j} \quad r = \lambda\, x.\, \mu\ ih.\ x\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n}}{\mu\ ih.\ (c_j\ \overline{s})\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..n} \rightsquigarrow \overline{[s/a_j]}[r/ih]t_j}$$

The elaboration of *this*, e_2, is formed by

$$\Gamma \vdash \overline{[s/a_j]}[r/ih]t_j \hookrightarrow \overline{[s'/a_j]}[r'/ih]t_j{}'$$

if $\Gamma \vdash r \hookrightarrow r'$. This follows by appealing to the rule for elaboration of untyped μ-expressions in Figure 17, revealing that

$$r' = \lambda\, x.\, |\texttt{mu}|\ x\ (\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n})$$

Now, note that by the definition of `mu` in Figure 9 that e_1 β-reduces to

$$\texttt{induction}\ (\lambda\, o.\, \lambda\, ih'.\, \lambda\, fr.\, (\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n})\ ih'\ fr)\ (c_j^{\text{FIX}}\ \overline{s'})$$

Which futher simplifies (under the λs of the first argument, and α-conversion of ih' to ih) to

$$\texttt{induction}\ (\lambda\, o.\, \lambda\, ih.\, \lambda\, fr.\, fr\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n})\ (c_j^{\text{FIX}}\ \overline{s'})$$

Now, recall that $c_j^{\text{FIX}}\ \overline{s'}$ is (by rules of Figure 14) definitionally equal to $\texttt{in}\ (c_j^{\text{FI}}\ \overline{s'})$, meaning that we can apply Requirement 2: there exists some λ-expression *ind* such that $\texttt{induction}\ (\lambda\, o.\, \lambda\, ih.\, \lambda\, fr.\, fr\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n}) \rightsquigarrow^* ind$, and furthermore that the expression we're working on reduces to

$$(\lambda\, o.\, \lambda\, ih.\, \lambda\, fr.\, fr\ (\lambda\, \overline{a_i}.\, t_i')_{i=1..n})\ \texttt{out}\ ind\ (c_j^{\text{FI}}\ \overline{s'})$$

By simple β-reduction (noting that o is not free in any t_i' or in *ind*) to

$$(c_j^{\text{FI}}\ \overline{s'})\ (\lambda\, \overline{a_k}.\, [ind/ih]t_k')_{k=1..n}$$

Unfolding, erasing, and reducing the definition of c_j^{FIX} we get $\lambda\, \overline{x_i}.\, x_j\ \overline{s}$. By noting that by the assumed derivation of reduction for the μ-expression the number of cases and expected arguments for each is as expected, we finally have a complete reduction sequence for

$$e_1 \rightsquigarrow^* \overline{[s'/a_j]}[ind/ih]t_j'$$

This is the form we want *this* term (e_1) to be in. For the other term (e_2), the argument is shorter. Recall that we established that the shape of our other term was

$$e_2 = \overline{[s'/a_j]}[\lambda\, x.\, \texttt{mu}\ x\ (\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})/ih]t'_j$$

By the definition of `mu` and η-contraction of the innermost substitution, this reduces to

$$\overline{[s'/a_j]}[\texttt{induction}\ (\lambda\, ih.\, \lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})/ih]t'_j$$

By our previous invocation of Requirement 2, we know further that this reduces to

$$\overline{[s'/a_j]}[ind/ih]t'_j$$

as required.

Case μ': By a similar argument to the case above. We go through the first few steps in case you are curious about the funny business of `out` disappearing in the reduction of μ. μ' allows one to "magically summon" an `out` of suitable type from the ether during induction

By the untyped elaboration rule for μ'-expressions in Figure 17, we know that

$$e_1 = |\texttt{mu'}|\ (c_j^{\text{FIX}}\ \overline{s})\ (\lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})$$

(where each $\overline{s'_i}$ and t'_i are the elaboration of $\overline{s_i}$, t_i)

By inversion of the reduction rule for μ' we have the form of the expression that our μ'-expression steps to is forced by the inference rule

$$\frac{1 \leq j \leq n \quad \#\overline{s} = \#\overline{a_j}}{\mu'\ (c_j\ \overline{s})\ \{c_i\ \overline{a_i} \to t_i\}_{i=1..n} \rightsquigarrow \overline{[s/a_j]}t_j}$$

By assumption, we have that the elaboration of *this*, e_2, is formed by

$$\Gamma \vdash \overline{[s/a_j]}t_j \hookrightarrow \overline{[s'/a_j]}t'_j$$

Where $\overline{s}$ elaborate to $\overline{s'}$ and t_j elaborates to t'_j. Now, note that by the definition of `mu'` in Figure 9 that e_1 β-reduces to

$$(\lambda\, x.\, x\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n})\ (\texttt{elimView out}\ (c_j^{\text{FIX}}\ \overline{s}))$$

By the definition of `elimView`, the expression `elimView out` reduces to `out`. By the rules of Figure 14, $c_j^{\text{FIX}}\ \overline{s'}$ unrolls to `in` $(c_j^{\text{FI}}\ \overline{s'})$. By Requirement 1 (`lambek1`), `out` composed with `in` of $(c_j^{\text{FI}}\ \overline{s'})$ β-reduces to just $(c_j^{\text{FI}}\ \overline{s'})$. So now we get that the whole expression reduces to

$$(c_j^{\text{FI}}\ \overline{s'})\ (\lambda\, \overline{a_i}.\, t'_i)_{i=1..n}$$

By a similar argument to for the μ-case above, we can conclude this reduces to

$$\overline{[s'/a_j]t'_j}$$

as required. □

Corolary 6.1.1 (Generalization of Theorem 6.1)**.**

- *If* $\Gamma \vdash p_1 \hookrightarrow p'_1$, $\Gamma \vdash p_2 \hookrightarrow p'_2$, *and* $|p_1| \cong |p_2|$, *then* $|p'_1| \cong_{\beta\eta} |p'_2|$

Proof. By a straightforward induction on the assumed derivations, appealing to normal $\beta\eta$-confluence and Theorems 6.1, 5.1, and 5.2. □

6.2 Type Preservation

Theorem 6.2 (Soundness: Type-Preservation)**.** *If* $\vdash \Gamma \hookrightarrow \Gamma'$ *then:*

- *If* $\Gamma \vdash K \hookrightarrow K'$ *then* $\Gamma^G, \Gamma' \vdash K'$
- *If* $\Gamma \vdash T : K \hookrightarrow T'$ *then some* K' *s.t.* $\Gamma^G, \Gamma' \vdash K'$ *and* $\Gamma^G, \Gamma' \vdash T' : K'$
- *If* $\Gamma \vdash t : T \hookrightarrow t'$ *then some* T' *s.t.* $\Gamma \vdash T : \star \hookrightarrow T'$ *and* $\Gamma \vdash t' : T'$

Proof. By mutual induction on the assumed derivation.

Term: $\Gamma \vdash t : T \hookrightarrow t'$ **[Mu']** By inversion, the assumed typing derivation is

$$\frac{\begin{array}{c}\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma,\ \mathcal{E}(D) = \texttt{Fix} \cdot D^{\mathrm{FI}}\ pos, \texttt{to/}D \in \Theta \\ \Gamma \vdash t : T \hookrightarrow t' \quad \Gamma \vdash T : \star \hookrightarrow T' \quad \Gamma \vdash P : D \to \star \hookrightarrow P' \\ \Gamma \vdash \{c_i\ \overline{a_i} \to t_i\} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}, T, is) \hookrightarrow (\{\overset{\lambda}{\Lambda}\ \overline{a_i}.\, t_i'\}, is')\end{array}}{\begin{array}{l}\mu'\texttt{<}is\texttt{>}\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : P\ (\texttt{to/}D\ \texttt{-}is\ t) \hookrightarrow \\ \texttt{mu'} \cdot D^{\mathrm{FI}}\ \texttt{-}pos\ \texttt{-}is'\ t' \cdot P'\ (\lambda\, x.\ x.2 \cdot (\texttt{Lift}_D \cdot P'\ \cdot T'\ is')\ (\overset{\lambda}{\Lambda}\ \overline{a_i}.\, \Lambda\, m.\, \Lambda\, eq.\ t_i')_{i=1..\#\Delta}\ \texttt{-}x\ \texttt{-}\beta)\end{array}}\ [\text{Mu'}]$$

We may further invert the the judgment (Cases) in the last premise of the rule, to get the following additional typing derivation (with some redundancy of premises betweem them):

$$\frac{\begin{array}{c}\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma \quad \texttt{Is/}D \in \Theta, \texttt{to/}D \in \Theta \quad \Gamma \vdash is : \texttt{Is/}D \cdot T \hookrightarrow is' \\ (c_i :\overset{\Pi}{\forall}\ \overline{a_i : A_i}.D \in \Delta)_{i=1..\#\Delta} \\ (\Gamma; \texttt{to/}D\ \texttt{-}is \vdash \overline{a_i : [T/R]A_i} \leqslant \overline{a_i : [D/R]A_i} \hookrightarrow \overline{s_i})_{i=1..\#\Delta} \\ (\Gamma \vdash \overset{\lambda}{\Lambda}\ \overline{a_i}.\, t_i : \overset{\Pi}{\forall}\ \overline{a_i : [T/R]A_i}.\, P\ (c_i\ \overline{s_i}) \hookrightarrow \overset{\lambda}{\Lambda}\ \overline{a_i}.\, t_i')_{i=1..\#\Delta}\end{array}}{\Gamma \vdash \{c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}_{i=1..\#\Delta}, T, is) \hookrightarrow (\{\overset{\lambda}{\Lambda}\ \overline{a_i}.\, t_i'\}_{i=1..\#\Delta}, is')}\ [\text{Cases}]$$

and we also have $\vdash \Gamma \hookrightarrow \Gamma'$.

By use of the IH on the premises of the derivations

- $\Gamma \vdash T : \star \hookrightarrow T'$
  yields $\Gamma^{\mathrm{G}}, \Gamma' \vdash T' : \star$ by mutual induction on and inversion of the kinding derivation
- $\Gamma \vdash is : \texttt{Is/}D \cdot T \hookrightarrow is'$
  yields $\Gamma^{\mathrm{G}}, \Gamma' \vdash is' : \texttt{IsD} \cdot D^{\mathrm{FI}}\ pos\ \cdot T'$ by induction on and inversion of the typing derivation of is
- $\Gamma \vdash P : D \to \star \hookrightarrow P'$
  yields $\Gamma^{\mathrm{G}}, \Gamma' \vdash P' : \texttt{Fix} \cdot F\ pos \to \star$ by mutual induction on and inversion of the kinding derivation of P
- $(c_i :\overset{\Pi}{\forall}\ \overline{a_i : A_i}.D \in \Delta)_{i=1..\#\Delta}$
  yields $(\Gamma^{\mathrm{G}}, \Gamma' \vdash \mathcal{E}(c_i) : \overset{\Pi}{\forall}\ \overline{a_i : [\mathcal{E}(D)/R]A_i'}.\, \mathcal{E}(D))_{i=1..\#\Delta}$ by Theorem 5.1, whose assumptions are satisfied:
  - by $\mathrm{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma$, which can only happen by rule [Data] (Figure 14)
  - by $(\Gamma, X{:}\star, R{:}\star \vdash \overset{\Pi}{\forall}\ \overline{a_i : A_i}.\, X : \star \hookrightarrow \overset{\Pi}{\forall}\ \overline{a_i : A_i'}.\, X$ implies $\Gamma^{\mathrm{G}}, \Gamma', R{:}\star, X{:}\star \vdash \overset{\Pi}{\forall}\ \overline{a_i : A_i'}.\, X : \star)_{i=1..\#\Delta}$
    From mutual induction (clearly $\Gamma, X{:}\star, R{:}s \hookrightarrow \Gamma', X{:}\star, R{:}\star$) and inversion of elaborating type inference rules

  and we may derive $(\Gamma \vdash c_i : \overset{\Pi}{\forall}\ \overline{a_i : [D/R]A_i}.\, D)_{i=1..\#\Delta}$ from rule [C] (Figure 16c)
- $(\Gamma; \texttt{to/}D\ \texttt{-}is \vdash \overline{a_i : [T/R]A_i} \leqslant \overline{a_i : [D/R]A_i} \hookrightarrow \overline{s_i})_{i=1..\#\Delta}$
  - yields $(\Gamma; (\overline{a_i : [T/R]A_i}) \vdash \overline{s_i} : (\overline{a_i : [D/R]A_i}) \hookrightarrow \overline{s_i'})_{i=1..\#\Delta}$ and $|\overline{s_i}| \cong |\overline{a_i}|$ by Theorem 5.3
    (by value-preservation (Theorem 6.1) $|\overline{s_i'}| \cong |\overline{a_i}|$)
  - yields $\Gamma^{\mathrm{G}}, \Gamma'; \texttt{toD}\ \texttt{-}pos\ is' \vdash \overline{a_i : [T/R]A_i'} \leqslant \overline{a_i : [\texttt{Fix} \cdot D^{\mathrm{FI}}\ pos/R]A_i'} \hookrightarrow \overline{s_i'}$ by Corollary 5.4.1

– yields $(\Gamma \vdash \overset{\Pi}{\forall}\ \overline{a_i : [D/R]A_i}.\ P\ (c_i\ \overline{s_i}) : \star \hookrightarrow \overset{\Pi}{\forall}\ \overline{a_i : [\texttt{Fix} \cdot D^{\mathrm{FI}}\ pos/R]A'_i}.\ P'\ (\mathcal{E}(c_i)\ \overline{s'_i}))$
the derivation of which we can easily derive from the pieces we have elaborated already

- $(\Gamma \vdash \overset{\lambda}{\Lambda}\ \overline{a_i}.\ t_i : \overset{\Pi}{\forall}\ \overline{a_i : [T/R]A_i}.\ P\ (c_i\ \overline{s_i}) \hookrightarrow \overset{\lambda}{\Lambda}\ \overline{a_i}.\ t'_i)_{i=1..\#\Delta}$
yields $(\Gamma^{\mathrm{G}}, \Gamma' \vdash \overset{\lambda}{\Lambda}\ \overline{a_i}.\ t_i' : \overset{\Pi}{\forall}\ \overline{a_i : [T'/R]A'_i}.\ P'\ (\mathcal{E}(c_i)\ \overline{s'_i}))_{i=1..\#\Delta}$ by induction on and inversion of the assumed typing derivation

The goal: show the elaborated term has type P' $(\texttt{toD} \cdot D^{\mathrm{FI}}\ \text{-}pos\ \text{-}is'\ t')$ (it is clear that this type is the elaboration of P $(\texttt{to}/D\ \text{-}is\ t)$ and has kind $\star$)

To show $\Gamma^{\mathrm{G}}, \Gamma' \vdash \texttt{mu'} \cdot D^{\mathrm{FI}}\ \text{-}is'\ t' \cdot P'\ (\lambda\, x.\ x.2 \cdot (\texttt{Lift}_D \cdot P' \cdot T'\ is')\ (\overset{\lambda}{\Lambda}\ \overline{a_i}.\ \Lambda\, m.\ \Lambda\, eq.\ t'_i)_{i=1..\#\Delta}\ \text{-}x\ \text{-}\beta) : P'\ (\texttt{toD}\ \text{-}pos\ is'\ t')$, we walk the sequence of arguments to `mu'`, typechecking as we go

- $\Gamma^{\mathrm{G}}, \Gamma' \vdash D^{\mathrm{FI}} : \star \to \star$
By Theorem 5.1
- $\Gamma^{\mathrm{G}}, \Gamma' \vdash pos : \texttt{Mono} \cdot D^{\mathrm{FI}}$
As above
- $\Gamma^{\mathrm{G}}, \Gamma' \vdash is' : \texttt{IsD} \cdot T'$
Established above
- $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : T'$
Established above
- $\Gamma^{\mathrm{G}}, \Gamma' \vdash P' : \texttt{Fix} \cdot D^{\mathrm{FI}}\ pos \to \star$
Established above
- $\Gamma^{\mathrm{G}}, \Gamma' \vdash \lambda\, x.\ x.2 \cdot (\texttt{Lift}_D \cdot P' \cdot T'\ is')\ (\overset{\lambda}{\Lambda}\ \overline{a_i}.\ \Lambda\, m.\ \Lambda\, eq.\ t'_i)_{i=1..\#\Delta}\ \text{-}x\ \text{-}\beta : \texttt{ByCases} \cdot P \cdot T^{\mathrm{R}\prime}\ is'$
If we can show this, then the type of the entire `mu'` application is P' $(\texttt{toD}\ \text{-}is'\ t')$, as required
  - Assume $x : D^{\mathrm{FI}} \cdot T'$, type the body
  - The head $x.2$ has type $\forall\, X : D^{\mathrm{F}} \cdot T' \to \star.\ \Pi\, x_i : (\overset{\Pi}{\forall}\ \overline{a_i : [T'/R]A'_i}.\ X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\ X\ x.1$
  By Theorem 5.1
  - Check $\texttt{Lift}_D \cdot P' \cdot T'\ is'$ has kind $D^{\mathrm{F}} \cdot T' \to \star$ – trivial
  - We must now check the remaining arguments can be given to a function of type $\Pi\, x_i : (\overset{\Pi}{\forall}\ \overline{a_i : [T'/R]A'_i}.\ (\texttt{Lift}_D \cdot P' \cdot T'\ is')\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\ (\texttt{Lift}_D \cdot P' \cdot T'\ is')\ x.1$
  - For each case $i = 1..\#\Delta$:
    * We know already $\Gamma^{\mathrm{G}}, \Gamma' \vdash \overset{\lambda}{\Lambda}\ \overline{a_i}.\ t'_i : \overset{\Pi}{\forall}\ \overline{a_i : [T/R]A'_i}.\ P'\ (c^{\mathrm{FIX}}\ \overline{s'_i})$ (where $c_i^{\mathrm{FIX}} = \mathcal{E}(c_i)$)
    * We want to show that this means $\Gamma^{\mathrm{G}}, \Gamma' \vdash \overset{\lambda}{\Lambda}\ \overline{a_i}.\ \Lambda\, m.\ \Lambda\, eq.\ t'_i : \overset{\Pi}{\forall}\ \overline{a_i : [T'/R]A'_i}.\ (\texttt{Lift}_D \cdot P' \cdot T'\ is')\ (c_i^{\mathrm{F}}\ \overline{a_i})$
    * Assume $\overline{a_i : [T'/R]A_i}$, $m : D^{\mathrm{FI}} \cdot T'$, $eq : \{m \simeq c_i^{\mathrm{F}}\ \overline{a_i}\}$, show t'_i: P' $(\texttt{in}\ \text{-}pos\ (\varphi\ eq - (\texttt{toFD}\ \text{-}pos\ \text{-}is'\ m)\ \{c^{\mathrm{F}}\ \overline{a_i}\}))$
    * By weakening of the current context, we know that t'_i has type P' $(\texttt{in} \cdot D^{\mathrm{FIX}}\ \text{-}pos\ (c_i^{\mathrm{FI}}\ \overline{s'_i}))$
    But these types are convertible! $\overline{s'_i}$ is convertible with $\overline{a_i}$, and also $(c_i^{\mathrm{FI}}\ \overline{a_i}) \cong c^{\mathrm{F}}\ \overline{a_i}$ (by inspection of the rule [cFI] of Figure 14)
  - The last two arguments to $x.2$ are easy: x has type $D^{\mathrm{FI}} \cdot T'$ as required, and β proves that $x.1$ (the term over which $x.2$ can prove properties) is indeed equal modulo erasure to x

Term: $\Gamma \vdash t : T \hookrightarrow t'$ **[Mu]** By inversion, our assumed typing derivation is

$$\frac{\begin{array}{c}\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma, \mathcal{E}(D) = \mathtt{Fix}\cdot D^{\mathrm{FI}}\ \mathit{pos}, \mathtt{to/}D \in \Theta \quad \Gamma \vdash P : D \to \star \hookrightarrow P' \quad \Gamma \vdash t : D \hookrightarrow t' \\ \Gamma' = \Gamma, \mathtt{Type/}\mathit{ih} : \star, \mathtt{isType/}\mathit{ih} : \mathtt{Is}/D \cdot \mathtt{Type/}\mathit{ih}, \mathit{ih} : \Pi\ y{:}\mathtt{Type/}\mathit{ih}.\ P\ (\mathtt{to}/D\ \text{-}\mathtt{isType/}\mathit{ih}\ y) \\ \Gamma' \vdash \{c_i\ \overline{a_i} \to t_i\} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}, \mathtt{Type/}\mathit{ih}, \mathtt{isType/}\mathit{ih}) \hookrightarrow (\{^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i{}'\}, \mathtt{isType/}\mathit{ih})\end{array}}{\begin{array}{c}\Gamma \vdash \mu\ \mathit{ih}.\ t\ @P\ \{\ c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : P\ t \hookrightarrow \\ \mathtt{mu}\cdot D^{\mathrm{FI}}\ \text{-}\mathit{pos}\ t'\cdot P'\ (\Lambda\,\mathtt{Type/}\mathit{ih}.\ \Lambda\,\mathtt{isType/}\mathit{ih}.\ \lambda\,\mathit{ih}.\ \lambda\,x. \\ x.2\cdot(\mathtt{Lift}_D\cdot P'\cdot \mathtt{Type/}\mathit{ih}\ \mathtt{isType/}\mathit{ih}\ (^{\lambda}_{\Lambda}\ \overline{a_i}.\ \Lambda\,m.\ \Lambda\,\mathit{eq}.\ t_i')_{i=1..\#\Delta}\ \text{-}x\ \text{-}\beta)\end{array}}\ [\mathrm{Mu}]$$

We may further invert the the judgment (Cases) in the last premise of the rule, to get the following additional typing derivation (with some redundancy of premises betweem them):

$$\frac{\begin{array}{c}\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma \quad \mathtt{Is}/D \in \Theta, \mathtt{to}/D \in \Theta \quad \Gamma \vdash \mathit{is} : \mathtt{Is}/D \cdot T \hookrightarrow \mathit{is}' \\ (c_i :^{\Pi}_{\forall}\ \overline{a_i : A_i}.D \in \Delta)_{i=1..\#\Delta} \\ (\Gamma; \mathtt{to}/D\ \text{-}\mathit{is} \vdash \overline{a_i : [T/R]A_i} \leqslant \overline{a_i : [D/R]A_i} \hookrightarrow \overline{s_i})_{i=1..\#\Delta} \\ (\Gamma \vdash {}^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i : {}^{\Pi}_{\forall}\ \overline{a_i : [T/R]A_i}.\ P\ (c_i\ \overline{s_i}) \hookrightarrow {}^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i')_{i=1..\#\Delta}\end{array}}{\Gamma \vdash \{c_i\ \overline{a_i} \to t_i\}_{i=1..\#\Delta} : \mathrm{Cases}(\{P\ (c_i\ \overline{s_i})\}_{i=1..\#\Delta}, T, \mathit{is}) \hookrightarrow (\{^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i{}'\}_{i=1..\#\Delta}, \mathit{is}')}\ [\mathrm{Cases}]$$

and we have also $\vdash \Gamma \hookrightarrow \Gamma''$ (to avoid a name clash with the extended context Γ' in the two derivations)

The proof proceeds similarly for the cases for **[Mu']** above. By our assumptions and by use of the IH on the premises of the derivation, we have

- $\Gamma \vdash t : D \hookrightarrow t'$
  yields $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : \mathtt{Fix}\cdot D^{\mathrm{FI}}\ \mathit{pos}$ by induction and by inversion of the typing derivation
- $\Gamma \vdash P : D \to \star \hookrightarrow P'$
  yields $\Gamma^{\mathrm{G}}, \Gamma' \vdash P' : \mathtt{Fix}\cdot F\ \mathit{pos} \to\to \star$ by mutual induction on and inversion of the kinding derivation of P
- $(c_i :^{\Pi}_{\forall}\ \overline{a_i : A_i}.D \in \Delta)_{i=1..\#\Delta}$
  yields $(\Gamma^{\mathrm{G}}, \Gamma'' \vdash \mathcal{E}(c_i) : {}^{\Pi}_{\forall}\ \overline{a_i : [\mathcal{E}(D)/R]A_i'}.\ \mathcal{E}(D))_{i=1..\#\Delta}$ by Theorem 5.1, whose assumptions:
  - $\mathrm{IndEl}[D,R,\Delta,\Theta,\mathcal{E}] \in \Gamma$, satisfied because (by inversion) can only happen by rule [Data] (Figure 14)
  - $(\Gamma, X{:}\star, R{:}\star \vdash {}^{\Pi}_{\forall}\ \overline{a_i : A_i}.\ X : \star \hookrightarrow {}^{\Pi}_{\forall}\ \overline{a_i : A_i'}.\ X$ implies $\Gamma^{\mathrm{G}}, \Gamma'', R{:}\star, X{:}\star \vdash {}^{\Pi}_{\forall}\ \overline{a_i : A_i'}.\ X : \star)_{i=1..\#\Delta}$
    satisfied by mutual induction (which we can apply because clearly $\Gamma, X{:}\star, R{:}\star \hookrightarrow \Gamma', X{:}\star, R{:}\star$) and inversion of elaborating type inference rules

  and we may derive $(\Gamma \vdash c_i : {}^{\Pi}_{\forall}\ \overline{a_i : [D/R]A_i}.\ D)_{i=1..\#\Delta}$ from rule [C] (Figure 16c)
- $\vdash \Gamma \hookrightarrow \Gamma''$
  From which $\vdash \Gamma' \hookrightarrow \Gamma'', \mathtt{Type/}\mathit{ih} : \star, \mathtt{isType/}\mathit{ih} : \mathtt{IsD}\cdot\mathtt{Type/}\mathit{ih}, \mathit{ih} : \Pi\ y : \mathtt{Type/}\mathit{ih}.\ P'\ (\mathtt{toD}\ \text{-}\mathit{pos}\ \text{-}\mathtt{isType/}\mathit{ih}\ y)$ follows from the above and use of Theorem 5.1. Call this extended elaborated context Γ'''
- $(\Gamma'; \mathtt{to}/D\ \text{-}\mathtt{isType/}\mathit{ih} \vdash \overline{a_i : [\mathtt{Type/}\mathit{ih}/R]A_i} \leqslant \overline{a_i : [D/R]A_i} \hookrightarrow \overline{s_i})_{i=1..\#\Delta}$
  - yields $(\Gamma'; (\overline{a_i : [\mathtt{Type/}\mathit{ih}/R]A_i}) \vdash \overline{s_i} : (\overline{a_i : [D/R]A_i}) \hookrightarrow \overline{s_i'})_{i=1..\#\Delta}$ and $|\overline{s_i}| \cong |\overline{a_i}|$ by Theorem 5.3
    (by value-preservation (Theorem 6.1) $|\overline{s_i'}| \cong |\overline{a_i}|$)
  - yields $\Gamma^{\mathrm{G}}, \Gamma'''; \mathtt{toD}\ \text{-}\mathit{pos}\ \mathtt{isType/}\mathit{ih} \vdash \overline{a_i : [\mathtt{Type/}\mathit{ih}/R]A_i'} \leqslant \overline{a_i : [\mathtt{Fix}\cdot D^{\mathrm{FI}}\ \mathit{pos}/R]A_i'} \hookrightarrow \overline{s_i'}$ by Corollary 5.4.1
  - yields $(\Gamma' \vdash {}^{\Pi}_{\forall}\ \overline{a_i : [D/R]A_i}.\ P\ (c_i\ \overline{s_i}) : \star \hookrightarrow {}^{\Pi}_{\forall}\ \overline{a_i : [\mathcal{E}(D)/R]A_i'}.\ P\ (\mathcal{E}(c_i)\ \overline{s_i'}))$
    the derivation of which we can easily obtain from the pieces we have elaborated already
- $(\Gamma' \vdash {}^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i : {}^{\Pi}_{\forall}\ \overline{a_i : [\mathtt{Type/}\mathit{ih}\ /\ R]A_i}.\ P\ (c_i\ \overline{s_i}) \hookrightarrow {}^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i')_{i=1..\#\Delta}$
  yields $(\Gamma^{\mathrm{G}}, \Gamma''' \vdash {}^{\lambda}_{\Lambda}\ \overline{a_i}.\ t_i{}' : {}^{\Pi}_{\forall}\ \overline{a_i : [\mathtt{Type/}\mathit{ih}\ /\ R]A_i'}.\ P'\ (\mathcal{E}(c_i)\ \overline{s_i'}))_{i=1..\#\Delta}$ by induction on and inversion of the assumed typing derivation

The goal

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash \mathtt{mu} \cdot D^{\mathrm{FI}}$ *-pos* $t' \cdot P'$ $(\Lambda\, \mathtt{Type/}ih.\, \Lambda\, \mathtt{isType/}ih.\, \lambda\, ih.\, \lambda\, x.$
  $x.2 \cdot (\mathtt{Lift}_D \cdot P' \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih\ (\overset{\lambda}{\Lambda}\, \overline{a_i}.\, \Lambda\, m.\, \Lambda\, eq.\, t'_i)_{i=1..\#\Delta}$ *-x -*$\beta) : P'\ t'$

We walk the sequence of arguments, typechecking as we go

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash D^{\mathrm{FI}} : \star \to \star$

  By Theorem 5.1

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash$ *pos* $: \mathtt{Mono} \cdot D^{\mathrm{FI}}$

  As above

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash t' : \mathtt{Fix} \cdot D^{\mathrm{FI}}$ *pos*

  Established from the assumptions

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash P' : D^{\mathrm{FIX}} \to \star$

  Established from the assumptions

- $\Gamma^{\mathrm{G}}, \Gamma'' \vdash \Lambda\, \mathtt{Type/}ih.\, \Lambda\, \mathtt{isType/}ih.\, \lambda\, ih.\, \lambda\, x.$
  $x.2 \cdot (\mathtt{Lift}_D \cdot P' \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih)\ (\overset{\lambda}{\Lambda}\, \overline{a_i}.\, \Lambda\, m.\, \Lambda\, eq.\, t'_i)$ *-x -*$\beta : \mathtt{ByInd} \cdot P'$

  If we can show this, then the entire `mu` expression has the type $P'\ t'$ as required.

  - Extending the context with the three abstracts $\mathtt{Type/}ih$, $\mathtt{isType/}ih$, ih and their expected types (given by `ByInd`) gives us precisely Γ'''.
  - Assume $x : D^{\mathrm{FI}} \cdot \mathtt{Type/}ih$, type the body
  - The head $x.2$ has type $\forall\, X : D^{\mathrm{F}} \cdot \mathtt{Type/}ih \to \star.\, \Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : [\mathtt{Type/}ih/R]A'_i}.\, X\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, X\ x.1$
    By Theorem 5.1
  - Check $\mathtt{Lift}_D \cdot P'\ \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih$ has kind $D^{\mathrm{F}} \cdot \mathtt{Type/}ih \to \star$ – trivial
  - We must now check the remaining arguments can be given to a function of type
    $\Pi\, x_i : (\overset{\Pi}{\forall}\, \overline{a_i : [\mathtt{Type/}ih/R]A'_i}.\, (\mathtt{Lift}_D \cdot P' \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih)\ (c_i^{\mathrm{F}}\ \overline{a_i}))_{i=1..\#\Delta}.\, (\mathtt{Lift}_D \cdot P'\ \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih)\ x.1$
  - For each case $i = 1..\#\Delta$:
    - We know already $\Gamma^{\mathrm{G}}, \Gamma''' \vdash \overset{\lambda}{\Lambda}\, \overline{a_i}.\, t'_i : \overset{\Pi}{\forall}\, \overline{a_i : [\mathtt{Type/}ih/R]A'_i}.\, P'\ (c^{\mathrm{FIX}}\ \overline{s'_i})$ (where $c_i^{\mathrm{FIX}} = \mathcal{E}(c_i)$)
    - We want to show that this means
      $\Gamma^{\mathrm{G}}, \Gamma''' \vdash \overset{\lambda}{\Lambda}\, \overline{a_i}.\, \Lambda\, m.\, \Lambda\, eq.\, t'_i : \overset{\Pi}{\forall}\, \overline{a_i : [\mathtt{Type/}ih/R]A'_i}.\, (\mathtt{Lift}_D \cdot P'\ \cdot \mathtt{Type/}ih\ \mathtt{isType/}ih)\ (c_i^{\mathrm{F}}\ \overline{a_i})$
    - Assume $\overline{a_i : [\mathtt{Type/}ih/R]A_i}$, $m : D^{\mathrm{FI}} \cdot \mathtt{Type/}ih$, $eq : \{m \simeq c_i^{\mathrm{F}}\ \overline{a_i}\}$,
      show t'_i: $P'\ (\mathtt{in}$ *-pos* $(\varphi\ eq - (\mathtt{toFD}$ *-pos* $\mathtt{-isType/}ih\ m)\ \{c^{\mathrm{F}}\ \overline{a_i}\}))$
    - By weakening of the current context, we know that t'_i has type $P'\ (\mathtt{in} \cdot D^{\mathrm{FIX}}$ *-pos* $(c_i^{\mathrm{FI}}\ \overline{s'_i}))$
      But these types are convertible! $\overline{s'_i}$ is convertible with $\overline{a_i}$, and as $(c_i^{\mathrm{FI}}\ \overline{a_i}) = c^{\mathrm{F}}\ \overline{a_i}$ (by inspection of the rule [cFI] of Figure 14)
  - The last two arguments to $x.2$ are easy: x has type $D^{\mathrm{FI}} \cdot \mathtt{Type/}ih$ as required, and β proves that $x.1$ (the term over which $x.2$ can prove properties) is indeed equal modulo erasure to x

Term: [c] [to] [is] By appeal to Theorem 5.1, appealing to the inductive hypothesis to show the elaborations of constructor argument types are well-kinded

Type: [D] [Is] By appeal to Theorem 5.1, appealing to the inductive hypothesis to show the elaborations of constructor argument types are well-kinded

Kind: Case

$$\frac{}{\Gamma \vdash \star \hookrightarrow \star}$$

Immediate

Kind: Case

$$\frac{\Gamma \vdash K_1 \hookrightarrow K_1' \quad \Gamma, X : K_1 \vdash K_2 \hookrightarrow K_2'}{\Gamma \vdash \Pi\, X : K_1.\, K_2 \hookrightarrow \Pi\, X : K_1'.\, K_2'}$$

- By the IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash K_1'$
- By the IH, $\Gamma^{\mathrm{G}}, \Gamma', X : K_1' \vdash K_2'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, X : K_1'.\, K_2'$

Kind: Case

$$\frac{\Gamma \vdash T : K_2 \hookrightarrow T' \quad \Gamma, x : T \vdash K_1 \hookrightarrow K_1'}{\Gamma \vdash \Pi\, x : T.\, K_1 \hookrightarrow \Pi\, x : T'.\, K_1'}$$

- By mutual induction, $\Gamma^{\mathrm{G}}, \Gamma' \vdash T' : K_2'$, for some K_2' where $\Gamma^{\mathrm{G}}, \Gamma' \vdash K_2'$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', x : T' \vdash K_1'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, x : T'.\, K_1'$

Type: Case

$$\frac{FV(p_1\ p_2) \subseteq dom(\Gamma) \quad \Gamma \vdash p_1 \hookrightarrow p_1' \quad \Gamma \vdash p_2 \hookrightarrow p_2'}{\Gamma \vdash \{p_1 \simeq p_2\} : \star \hookrightarrow \{p_1' \simeq p_2'\}}$$

- Appeal to Lemma 6.4 to get $FV(p_1'\ p_2') \subseteq dom(\Gamma^{\mathrm{G}}, \Gamma')$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \{p_1' \simeq p_2'\} : \star$

Type: Case

$$\frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star \hookrightarrow T_2'}{\Gamma \vdash \iota\, x : T_1.\, T_2 : \star \hookrightarrow \iota\, x : T_1'.\, T_2'}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash T_1' : \star$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', x : T_1' \vdash T_2' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \iota\, x : T_1'.\, T_2' : \star$

Type: Case

$$\frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star T_2'}{\Gamma \vdash \forall\, x : T_1.\, T_2 : \star \hookrightarrow \forall\, x : T_1'.\, T_2'}$$

Similar to above

Type: Case

$$\frac{\Gamma \vdash T_1 : \star \hookrightarrow T_1' \quad \Gamma, x : T_1 \vdash T_2 : \star \hookrightarrow T_2'}{\Gamma \vdash \Pi\, x : T_1.\, T_2 : \star \hookrightarrow \Pi\, x : T_1'.\, T_2'}$$

Similar to above

Type: Case

$$\frac{\Gamma \vdash K \hookrightarrow K' \quad \Gamma, X : K \vdash T : \star \hookrightarrow T'}{\Gamma \vdash \forall\, X : K.\, T : \star \hookrightarrow \forall\, X : K'.\, T'}$$

- By mutual induction, $\Gamma^{\mathrm{G}}, \Gamma' \vdash K'$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', X : K' \vdash T' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \forall\, X : K'.\, T' : \star$

Type: Case

$$\frac{\Gamma \vdash S : K_1 \hookrightarrow S' \quad \Gamma, x : S \vdash T : K_2 \hookrightarrow T'}{\Gamma \vdash \lambda\, x : S.\, T : \Pi\, x : S.\, K_2 \hookrightarrow \lambda\, x : S'.\, T'}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash S' : K_1'$ for some $K_1{}'$ where $\Gamma^{\mathrm{G}}, \Gamma' \vdash K_1'$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', x : S' \vdash T' : K_2'$ for some K_2' where $\Gamma^{\mathrm{G}}, \Gamma', X : S' \vdash K_2' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \lambda\, x : S'.\, T' : \Pi\, x : S'.\, K_2'$

Type: Case

$$\frac{\Gamma \vdash K_1 \hookrightarrow K_1' \quad \Gamma, X : K_1 \vdash T : K_2 \hookrightarrow T'}{\Gamma \vdash \lambda\, X : K_1.\, T : \Pi\, X : K_1.\, K_2 \hookrightarrow \lambda\, X : K_1'.\, T'}$$

Similar to above, appealing to mutual induction for the kind K_1 of the λ-bound variable X

Type: Case

$$\frac{\Gamma \vdash T_1 : \Pi\, X : K_2.\, K_1 \hookrightarrow T_1' \quad \Gamma \vdash T_2 : K_2 \hookrightarrow T_2'}{\Gamma \vdash T_1 \cdot T_2 : [T_2/X]K_1 \hookrightarrow T_1' \cdot T_2'}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash T_1' : \Pi\, K_2' : K_1.$ for some K_2', K_1' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, X : K_2'.\, K_1$ (inversion of kind elaboration)
- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash T_2' : K_2'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash T_1' \cdot T_2' : [T_2'/X]K_1'$

Type: Case

$$\frac{\Gamma \vdash T : \Pi\, x : S.\, K \hookrightarrow T' \quad \Gamma \vdash t : S \hookrightarrow t'}{\Gamma \vdash T\ t : [t/x]K \hookrightarrow T'\ t'}$$

Similar to above, appealing to mutual induction for the elaboration of term argument t

Type: Case

$$\frac{}{\Gamma \vdash X : \Gamma(X) \hookrightarrow X}$$

Immediate (Lemma 6.4 and Figure 12a)

Term: Case

$$\frac{}{\Gamma \vdash x : \Gamma(x) \hookrightarrow x}$$

Immediate (Lemma 6.4 and Figure 12a)

Term: Case

$$\frac{\Gamma \vdash S : \star \hookrightarrow S' \quad \Gamma, x : S \vdash t : T \hookrightarrow t'}{\Gamma \vdash \lambda\, x.\, t : \Pi\, x : S.\, T \hookrightarrow \lambda\, x.\, t'}$$

- By mutual induction, $\Gamma^{\mathrm{G}}, \Gamma' \vdash S' : \star$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', x : S' \vdash t' : T'$ for some T' s.t. $\Gamma^{\mathrm{G}}, \Gamma', x : S' \vdash T' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \lambda\, x.\, t' : \Pi\, x : S'.\, T'$

Term: Case

$$\frac{\Gamma \vdash K \hookrightarrow K' \quad X \notin FV(|t|) \quad \Gamma, X : K \vdash t : T \hookrightarrow t'}{\Gamma \vdash \Lambda\, X.\, t : \forall\, X : K.\, T \hookrightarrow \Lambda\, X.\, t'}$$

- By mutual induction, $\Gamma^{\mathrm{G}}, \Gamma' \vdash K'$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma', X : K' \vdash t' : T'$ for some T' such that $\Gamma^{\mathrm{G}}, \Gamma' \vdash T' : \star$
- By Lemma 6.4, $X \notin FV(|t'|)$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Lambda\, X.\, t' : \forall\, X : K'.\, T'$

Term: Case

$$\frac{x \notin FV(|t|) \quad \Gamma, x : S \vdash t : T}{\Gamma \vdash \Lambda\, x.\, t : \forall\, x : S.\, T}$$

Similar to above, invoking mutual induction for elaboration of type S of the Λ-bound variable

Term: Case

$$\frac{\Gamma \vdash t : \Pi\, x : S.\, T \hookrightarrow t' \quad \Gamma \vdash s : S \hookrightarrow s'}{\Gamma \vdash t\ s : [s/x]T \hookrightarrow t'\ s'}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : \Pi\, x : S'.\, T'$ for some S' and T' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \Pi\, x : S'.\, T' : \star$ (inversion of type elaboration)
- By IH $\Gamma^{\mathrm{G}}, \Gamma' \vdash s' : S'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash t'\ s' : [s'/x]T'$

Term: Case

$$\frac{\Gamma \vdash t : \forall\, X : K.\, T \hookrightarrow t' \quad \Gamma \vdash S : K \hookrightarrow S'}{\Gamma \vdash t \cdot S : [S/X]T \hookrightarrow t' \cdot S'}$$

Similar to above, invoking mutual induction for the the type argument S

Term: Case

$$\frac{\Gamma \vdash t : \forall\, x : S.\, T \hookrightarrow t' \quad \Gamma \vdash s : S \hookrightarrow s'}{\Gamma \vdash t\ \text{-}s : [s/x]T \hookrightarrow t'\ \text{-}s'}$$

Similar to above

Term: Case

$$\frac{\Gamma \vdash t : S \hookrightarrow t' \quad S \cong T \quad \Gamma \vdash T : \star \hookrightarrow T'}{\Gamma \vdash t : T \hookrightarrow t'}$$

By the IH, invoking Corollary 6.1.1 and referencing the type-convertibility relation of [Stu18]

Term: Case

$$\frac{\Gamma \vdash t_1 : T_1 \hookrightarrow t_1' \quad \Gamma \vdash t_2 : [t_1/x]T_2 \hookrightarrow t_2' \quad t_1' \cong t_2'}{\Gamma \vdash [t_1, t_2] : \iota\, x : T_1.\, T_2 \hookrightarrow [t_1', t_2']}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t_1' : T_1'$ for some T_1' s.t $\Gamma^{\mathrm{G}}, \Gamma' \vdash T_1' : \star$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t_2' : [t_1'/x]T_2'$ for some T_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma', x : T_1' \vdash T_2' : \star$
- By Corollary 6.1.1, $t_1' \cong t_2'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash [t_1', t_2'] : \iota\, x : T_1'.\, T_2'$

Term: Case

$$\frac{\Gamma \vdash t : \iota\, x{:}T_1.\, T_2 \hookrightarrow t'}{\Gamma \vdash t.1 : T_1 \hookrightarrow t'.1}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : \iota\, x{:}T_1'.\, T_2'$ for some T_1', T_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \iota\, x{:}T_1'.\, T_2' : \star$ (inversion of type elaboration rules)
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash t.1' : T_1'$

Term: Case

$$\frac{\Gamma \vdash t : \iota\, x{:}T_1.\, T_2 \hookrightarrow t'}{\Gamma \vdash t.2 : [t.1/x]T_2 \hookrightarrow t'.2}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : \iota\, x{:}T_1'.\, T_2'$ for some T_1', T_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \iota\, x{:}T_1'.\, T_2' : \star$ (inversion of type elaboration rules)
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash t'.2 : [t'.1/x]T_2'$

Term: Case

$$\frac{\Gamma \vdash \{|t| \simeq |t|\} : \star \hookrightarrow \{p \simeq p\}}{\Gamma \vdash \beta : \{|t| \simeq |t|\} \hookrightarrow \beta}$$

- By mutual induction $\Gamma^{\mathrm{G}}, \Gamma' \vdash \{p \simeq p\} : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \beta : \{p \simeq p\}$

Term: Case

$$\frac{\begin{array}{cc}\Gamma \vdash s : \{|t_1| \simeq |t_2|\} \hookrightarrow s' & \Gamma \vdash t : [t_1/x]T_1 \hookrightarrow t' \\ T_1 \cong T_2 & \Gamma \vdash [t_2/x]T_2 : \star \hookrightarrow [t_2'/x]T_2'\end{array}}{\Gamma \vdash \rho\ s\ @\ x.T_2\ \text{-}\ t : [t_2/x]T_2 \hookrightarrow \rho\ s'\ @\ x.T_2'\ \text{-}\ t'}$$

Since you have made it this far: there's actually quite a subtle issue in [Stu18] concerning ρ, namely that rewriting by untyped equalities may produce a result type that is not re-kindable. This does not effect the denotational semantics (nor soundness proof) for Cedille. We have separately developed kind-preserving rewrite rules for the unguided (no @ $x.T_2$) ρ; the rule here is a specificational one that is consistent with the original rules but requires additional burden on the programmer to provide the correctly-rewritten T_2 that satisfies the premises (being convertible with T_1 modulo erasure and being well-kinded when instantiating t_2 for x)

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash s' : \{|t_1'| \simeq |t_2'|\}$ where some t_1' and t_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \{|t_1'| \simeq |t_2'|\} : \star$ (inversion of type elaboration rules)
- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : [t_1'/x]T_1'$ for some T_1' such that $\Gamma^{\mathrm{G}}, \Gamma' \vdash [t_1'/x]T_1' : \star$
- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash [t_2'/x]T_2' : \star$ for some T_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash [t_2'/x]T_2' : \star$
- By Lemma 6.1.1 and reference to the type convertibility rules of [Stu18], $T_1' \cong T_2'$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \rho\ s'\ @\ x.T_2'\ -t' : [t_2'/x]T_2'$

Term: Case

$$\frac{\Gamma \vdash s : \{|t_1| \simeq |t_2|\} \hookrightarrow s' \quad \Gamma \vdash t_1 : T \hookrightarrow t_1' \quad \Gamma \vdash t_2 \hookrightarrow t_2'}{\Gamma \vdash \varphi\ s - t_1\ \{t_2\} : T \hookrightarrow \varphi\ s' - t_1'\ \{t_2'\}}$$

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash s' : \{|t_1'| \simeq |t_2'|\}$ for some t_1', t_2' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash \{|t_1'| \simeq |t_2'|\} : \star$ (inversion of type elaboration rules)
- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t_1' : T'$ for some T' s.t. $\Gamma^{\mathrm{G}}, \Gamma' \vdash T' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \varphi\ s' - t_1'\ \{t_2\} : T'$

Term: Case

$$\frac{\Gamma \vdash t : \{\lambda\, x.\, \lambda\, y.\, x \simeq \lambda\, x.\, \lambda\, y.\, x\} \hookrightarrow t' \quad \Gamma \vdash T : \star \hookrightarrow T'}{\Gamma \vdash \delta\ T - t : T \hookrightarrow \delta\ T' - t'}$$

Note that we are considering the restricted form of δ. Proving sound elaboration for the full Böhm-out algorithm, implemented by the Cedille tool, is beyond the scope of our contributions.

- By IH, $\Gamma^{\mathrm{G}}, \Gamma' \vdash t' : \{\lambda\, x.\, \lambda\, y.\, x \simeq \lambda\, x.\, \lambda\, y.\, y\}$
- By mutual induction, $\Gamma^{\mathrm{G}}, \Gamma' \vdash T' : \star$
- Conclude $\Gamma^{\mathrm{G}}, \Gamma' \vdash \delta\ T' - t' : T'$

□

6.3 Termination Guarantee

Theorem 6.3 (Call-by-name Normalization)**.** *Suppose* $\Gamma \vdash t : D \hookrightarrow t'$ *and* $\mathit{IndEl}[D, R, \Delta, \Theta, \mathcal{E}] \in \Gamma$*. Suppose further that* t *is closed. Then* $|t'|$ *is call-by-name normalizing.*

Proof. By Theorem 6.2, there is a type T such that $\Gamma \vdash D : \star \hookrightarrow T$ and $\Gamma \vdash t' : T'$. By inversion on type elaboration, T is $\mathcal{E}(D)$, which is of the form $\texttt{Fix} \cdot D^{\mathrm{FI}}$ *pos*. By Requirement 3, there is an identity function from this type to a Π-type. So, by Theorem 4 of [Stu18], $|t'|$ is call-by-name normalizing. □

6.4 Additional Proofs

Lemma 6.4.

- *If* $\Gamma \vdash K \hookrightarrow K'$ *then* $FV(K) = FV(K')$
- *If* $\Gamma \vdash T : K \hookrightarrow T'$ *then* $FV(T) = FV(T')$
- *If* $\Gamma \vdash t : T \hookrightarrow t'$ *then* $FV(t) = FV(t')$
- *If* $\vdash \Gamma \hookrightarrow \Gamma'$ *then* $DV(\Gamma) = DV(\Gamma') \backslash DV(\Gamma^G)$

Proof. By a straightforward mutual induction on the assumed derivations. □

Lemma 6.4 is essential for type-checking the term elaborations of implicit products and kind-checking the elaborations of equality types.

References

[DFS18] Larry Diehl, Denis Firsov, and Aaron Stump. Generic zero-cost reuse for dependent types. *Proc. ACM Program. Lang.*, 2(ICFP):104:1–104:30, July 2018.

[FDJS18] Denis Firsov, Larry Diehl, Christopher Jenkins, and Aaron Stump. Course-of-value induction in Cedille, 2018.

[Lam68] Joachim Lambek. A fixpoint theorem for complete categories. *Mathematische Zeitschrift*, 103(2):151–161, 1968.

[Stu18] Aaron Stump. Syntax and semantics of Cedille, 2018.