Gradual Intersection Types

Pedro Ângelo, Mário Florido

March 1, 2018

1 Language Definition

Syntax

$$
\begin{aligned}
&Types\ \ I ::= \ Int \mid Bool \mid Dyn \mid I \rightarrow T \mid I \cap \ldots \cap I \\
&\qquad\ \ T ::= \ Int \mid Bool \mid Dyn \mid T \rightarrow T \\
&Ground\ Types\ G\ ::= \ Int \mid Bool \mid Dyn \rightarrow Dyn \\
&Casts\ \ c\ ::= c : T \Rightarrow^{l} T\ ^{cl} \mid blame\ T\ T\ l\ ^{cl} \mid \varnothing\ T\ ^{cl} \\
&Expressions\ \ e\ ::= x \mid \lambda x : I\ .\ e \mid e\ e \mid n \mid true \mid false \\
&\qquad\qquad\qquad\quad \mid e : c \cap \ldots \cap c \mid blame_{I}\ l \\
&Cast\ Values\quad cv\ ::= cv1 \mid cv2 \\
&\qquad\qquad\quad cv1\ ::= \varnothing\ T\ ^{cl} : G \Rightarrow^{l} Dyn\ ^{cl} \\
&\qquad\qquad\qquad\quad \mid cv1 : G \Rightarrow^{l} Dyn\ ^{cl} \\
&\qquad\qquad\qquad\quad \mid \varnothing\ T\ ^{cl} : T_1 \rightarrow T_2 \Rightarrow^{l} T_3 \rightarrow T_4\ ^{cl} \\
&\qquad\qquad\qquad\quad \mid cv1 : T_1 \rightarrow T_2 \Rightarrow^{l} T_3 \rightarrow T_4\ ^{cl} \\
&\qquad\qquad\quad cv2\ ::= blame\ T\ T\ l\ ^{cl} \\
&\qquad\qquad\qquad\quad \mid \varnothing\ T\ ^{cl} \\
&Values\ \ v\ ::= x \mid \lambda x : I\ .\ e \mid n \mid true \mid false \mid blame_{I}\ l \\
&\qquad\qquad \mid v : cv_1 \cap \ldots \cap cv_n\ such\ that \\
&\qquad\qquad \neg(\forall_{i \in 1..n}\ .\ cv_i = blame\ T\ T\ l\ ^{cl}) \wedge \\
&\qquad\qquad \neg(\forall_{i \in 1..n}\ .\ cv_i = \varnothing\ T\ ^{cl})
\end{aligned}
$$

Figure 1: Gradual Intersection System

$\boxed{\Gamma \vdash_{\cap G} e : T}$ Typing

$$\frac{x : T \in \Gamma}{\Gamma \vdash_{\cap G} x : T}\ \text{T-Var} \qquad \frac{\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap G}\ e : T}{\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_1 \cap \ldots \cap T_n \to T}\ \text{T-Abs}$$

$$\frac{\Gamma, x : T_i \vdash_{\cap G}\ e : T}{\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_i \to T}\ \text{T-Abs'}$$

$$\frac{\begin{array}{cc}\Gamma \vdash_{\cap G} e_1 : PM & PM \rhd T_1 \cap \ldots \cap T_n \to T \\ \Gamma \vdash_{\cap G} e_2 : T_1' \cap \ldots \cap T_n' & T_1' \cap \ldots \cap T_n' \sim T_1 \cap \ldots \cap T_n\end{array}}{\Gamma \vdash_{\cap G} e_1\ e_2 : T}\ \text{T-App}$$

$$\frac{\Gamma \vdash_{\cap G} e : T_1\ \ldots\ \Gamma \vdash_{\cap G} e : T_n}{\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n}\ \text{T-Gen} \qquad \frac{\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n}{\Gamma \vdash_{\cap G} e : T_i}\ \text{T-Inst}$$

$$\frac{}{\Gamma \vdash_{\cap G} n : Int}\ \text{T-Int} \qquad \frac{}{\Gamma \vdash_{\cap G} true : Bool}\ \text{T-True} \qquad \frac{}{\Gamma \vdash_{\cap G} false : Bool}\ \text{T-False}$$

$\boxed{T \sim T}$ Consistency

$$B \sim B \qquad T \sim Dyn \qquad Dyn \sim T \qquad \frac{T_1 \sim T_3 \qquad T_2 \sim T_4}{T_1 \to T_2 \sim T_3 \to T_4} \qquad \frac{T_1 \sim T_1'\ \ldots\ T_n \sim T_n'}{T_1 \cap \ldots \cap T_n \sim T_1' \cap \ldots \cap T_n'}$$

$$\frac{T \sim T_1\ \ldots\ T \sim T_n}{T \sim T_1 \cap \ldots \cap T_n} \qquad \frac{T_1 \sim T\ \ldots\ T_n \sim T}{T_1 \cap \ldots \cap T_n \sim T}$$

$\boxed{T \rhd T}$ Pattern Matching

$$T_1 \to T_2 \rhd T_1 \to T_2 \qquad Dyn \rhd Dyn \to Dyn$$

Figure 2: Gradual Intersection Type System ($\vdash_{\cap G}$)

$\boxed{T \sqsubseteq T}$ Type Precision

$$Dyn \sqsubseteq T \qquad B \sqsubseteq B \qquad \frac{T_1 \sqsubseteq T_3 \qquad T_2 \sqsubseteq T_4}{T_1 \to T_2 \sqsubseteq T_3 \to T_4} \qquad \frac{T_1 \sqsubseteq T_1' \ \ldots \ T_n \sqsubseteq T_n'}{T_1 \cap \ldots \cap T_n \sqsubseteq T_1' \cap \ldots \cap T_n'}$$

$$\frac{T \sqsubseteq T_1 \ \ldots \ T \sqsubseteq T_n}{T \sqsubseteq T_1 \cap \ldots \cap T_n} \qquad \frac{T_1 \sqsubseteq T \ \ldots \ T_n \sqsubseteq T}{T_1 \cap \ldots \cap T_n \sqsubseteq T}$$

$\boxed{c \sqsubseteq c}$ Cast Precision

$$\frac{c \sqsubseteq c' \qquad T_1 \sqsubseteq T_1' \qquad T_2 \sqsubseteq T_2'}{c : T_1 \Rightarrow^l T_2 \ ^{cl} \sqsubseteq c' : T_1' \Rightarrow^{l'} T_2' \ ^{cl'}} \qquad \frac{c \sqsubseteq c' \qquad \vdash_{\cap IC} c' : T \qquad T_1 \sqsubseteq T \qquad T_2 \sqsubseteq T}{c : T_1 \Rightarrow^l T_2 \ ^{cl} \sqsubseteq c'}$$

$$\frac{c \sqsubseteq c' \qquad \vdash_{\cap IC} c : T \qquad T \sqsubseteq T_1 \qquad T \sqsubseteq T_2}{c \sqsubseteq c' : T_1 \Rightarrow^l T_2 \ ^{cl}} \qquad \frac{T_I \sqsubseteq T_I' \qquad T_F \sqsubseteq T_F'}{blame\ T_I\ T_F\ l\ ^{cl} \sqsubseteq blame\ T_I'\ T_F'\ l'\ ^{cl'}}$$

$$\frac{T \sqsubseteq T'}{\varnothing\ T\ ^{cl} \sqsubseteq \varnothing\ T'\ ^{cl'}}$$

$\boxed{e \sqsubseteq e}$ Expression Precision

$$x \sqsubseteq x \qquad \frac{T \sqsubseteq T' \qquad e \sqsubseteq e'}{\lambda x : T\ .\ e \sqsubseteq \lambda x : T'\ .\ e'} \qquad \frac{e_1 \sqsubseteq e_1' \qquad e_2 \sqsubseteq e_2'}{e_1\ e_2 \sqsubseteq e_1'\ e_2'} \qquad n \sqsubseteq n \qquad true \sqsubseteq true$$

$$false \sqsubseteq false \qquad \frac{e \sqsubseteq e' \qquad c_1 \sqsubseteq c_1' \ldots c_n \sqsubseteq c_n'}{e : c_1 \cap \ldots \cap c_n \sqsubseteq e' : c_1' \cap \ldots \cap c_n'}$$

$$\frac{e \sqsubseteq e' \qquad \Gamma \vdash_{\cap CC} e' : T \qquad \vdash_{\cap IC} c_1 : T_1 \ldots \vdash_{\cap IC} c_n : T_n \qquad T_1 \cap \ldots \cap T_n \sqsubseteq T}{e : c_1 \cap \ldots \cap c_n \sqsubseteq e'}$$

$$\frac{e \sqsubseteq e' \qquad \Gamma \vdash_{\cap CC} e : T \qquad \vdash_{\cap IC} c_1 : T_1 \ldots \vdash_{\cap IC} c_n : T_n \qquad T \sqsubseteq T_1 \cap \ldots \cap T_n}{e \sqsubseteq e' : c_1 \cap \ldots \cap c_n}$$

$$\frac{\Gamma \vdash_{\cap CC} e : T \qquad T \sqsubseteq T'}{e \sqsubseteq blame_{T'}\ l}$$

Figure 3: Precision ($\sqsubseteq$)

$\boxed{\Gamma \vdash_{\cap CC} e : T}$ Typing

$$\textit{static type system } (\Gamma \vdash_{\cap S} e : T) \textit{ rules and}$$

$$\frac{\Gamma \vdash_{\cap CC} e_1 : T_{11} \rightarrow T_{12} \cap \ldots \cap T_{n1} \rightarrow T_{n2} \qquad \Gamma \vdash_{\cap CC} e_2 : T_{11} \cap \ldots \cap T_{n1}}{\Gamma \vdash_{\cap CC} e_1\ e_2 : T_{12} \cap \ldots \cap T_{n2}} \text{ T-APP'}$$

$$\frac{\begin{array}{c}\Gamma \vdash_{\cap CC} e : T \qquad \vdash_{\cap IC} c_1 : T_1\ \ldots\ \vdash_{\cap IC} c_n : T_n \\ initialType(c_1) \cap \ldots \cap initialType(c_n) = T\end{array}}{\Gamma \vdash_{\cap CC} e : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n} \text{ T-INTERSECTIONCAST}$$

$$\frac{}{\Gamma \vdash_{\cap CC} blame_T\ l : T} \text{ T-BLAME}$$

$\boxed{\text{initialType(c)} = \text{T}}$

$$initialType(c : T_1 \Rightarrow^l T_2\ {}^{cl}) = initialType(c)$$

$$initialType(\varnothing\ T\ {}^{cl}) = T$$

$$initialType(blame\ T_I\ T_F\ l\ {}^{cl}) = T_I$$

$\boxed{\text{finalType(c)} = \text{T}}$

$$finalType(c : T_1 \Rightarrow^l T_2\ {}^{cl}) = T_2$$

$$finalType(\varnothing\ T\ {}^{cl}) = T$$

$$finalType(blame\ T_I\ T_F\ l\ {}^{cl}) = T_F$$

Figure 4: Intersection Cast Calculus ($\vdash_{\cap CC}$)

$\boxed{\Gamma \vdash_{\cap CC} e \rightsquigarrow e : T}$ Compilation

$$\frac{x : T \in \Gamma}{\Gamma \vdash_{\cap CC} x \rightsquigarrow x : T} \text{ C-VAR}$$

$$\frac{\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap CC} e \rightsquigarrow e' : T}{\Gamma \vdash_{\cap CC} (\lambda x : T_1 \cap \ldots \cap T_n \ . \ e) \rightsquigarrow (\lambda x : T_1 \cap \ldots \cap T_n \ . \ e') : T_1 \cap \ldots \cap T_n \rightarrow T} \text{ C-ABS}$$

$$\frac{\Gamma, x : T_i \vdash_{\cap CC} e \rightsquigarrow e' : T}{\Gamma \vdash_{\cap CC} (\lambda x : T_1 \cap \ldots \cap T_n \ . \ e) \rightsquigarrow (\lambda x : T_1 \cap \ldots \cap T_n \ . \ e') : T_i \rightarrow T} \text{ C-ABS'}$$

$$\frac{\begin{array}{c} \Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : PM \qquad PM \rhd T_1 \cap \ldots \cap T_n \rightarrow T \\ \Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e_2' : T_1' \cap \ldots \cap T_n' \qquad T_1' \cap \ldots \cap T_n' \sim T_1 \cap \ldots \cap T_n \qquad instances(PM) = S_1 \\ instances(T_1 \cap \ldots \cap T_n \rightarrow T) = S_2 \qquad instances(T_1' \cap \ldots \cap T_n') = S_3 \\ instances(T_1 \cap \ldots \cap T_n) = S_4 \qquad S_1,\ S_2,\ e_1' \hookrightarrow e_1'' \qquad S_3,\ S_4,\ e_2' \hookrightarrow e_2'' \end{array}}{\Gamma \vdash_{\cap CC} e_1 \ e_2 \rightsquigarrow e_1'' \ e_2'' : T} \text{ C-APP}$$

$$\frac{\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1 \ \ldots \ \Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_n}{\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1 \cap \ldots \cap T_n} \text{ C-GEN} \qquad \frac{\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1 \cap \ldots \cap T_n}{\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_i} \text{ C-INST}$$

$$\frac{}{\Gamma \vdash_{\cap CC} n \rightsquigarrow n : Int} \text{ C-INT} \qquad \frac{}{\Gamma \vdash_{\cap CC} true \rightsquigarrow true : Bool} \text{ C-TRUE}$$

$$\frac{}{\Gamma \vdash_{\cap CC} false \rightsquigarrow false : Bool} \text{ C-FALSE}$$

$\boxed{\text{instances(T)} = \{\text{T}\}}$

$$instances(Int) = \{Int\}$$

$$instances(Bool) = \{Bool\}$$

$$instances(Dyn) = \{Dyn\}$$

$$\frac{instances(T_1) = \{T_{11}, \ldots, T_{1n}\}}{instances(T_1 \rightarrow T_2) = \{T_{11} \rightarrow T_2, \ldots, T_{1n} \rightarrow T_2\}}$$

$$\frac{instances(T_1) = \{T_{11}, \ldots, T_{1m}\} \ldots instances(T_n) = \{T_{n1}, \ldots, T_{nj}\}}{instances(T_1 \cap \ldots \cap T_n) = \{T_{11}, \ldots, T_{1m}, \ldots, T_{n1}, \ldots, T_{nj}\}}$$

$\boxed{S,\ S,\ e \hookrightarrow e}$

$$\{T_1\},\ \{T_2\},\ e \hookrightarrow e : (\varnothing\ T_1\ ^0 : T_1 \Rightarrow^l T_2\ ^0)$$

$$\{T_{11}, \ldots, T_{1n}\},\ \{T_{21}, \ldots, T_{2n}\},\ e \hookrightarrow e : (\varnothing\ T_{11}\ ^0 : T_{11} \Rightarrow^{l_1} T_{21}\ ^0) \cap \ldots \cap (\varnothing\ T_{1n}\ ^0 : T_{1n} \Rightarrow^{l_n} T_{2n}\ ^0)$$

$$\{T_{11}, \ldots, T_{1n}\},\ \{T_2\},\ e \hookrightarrow e : (\varnothing\ T_{11}\ ^0 : T_{11} \Rightarrow^{l_1} T_2\ ^0) \cap \ldots \cap (\varnothing\ T_{1n}\ ^0 : T_{1n} \Rightarrow^{l_n} T_2\ ^0)$$

$$\{T_1\},\ \{T_{21}, \ldots, T_{2n}\},\ e \hookrightarrow e : (\varnothing\ T_1\ ^0 : T_1 \Rightarrow^{l_1} T_{21}\ ^0) \cap \ldots \cap (\varnothing\ T_1\ ^0 : T_1 \Rightarrow^{l_n} T_{2n}\ ^0)$$

Figure 5: Compilation to the Cast Calculus

$\boxed{e \longrightarrow_{\cap CC} e}$ Evaluation

Push blame to top level

$$\frac{\Gamma \vdash_{\cap CC} (blame_{T_2}\ l)\ e_2 : T_1}{(blame_{T_2}\ l)\ e_2 \longrightarrow_{\cap CC} blame_{T_1}\ l} \text{ E-PushBlame1}$$

$$\frac{\Gamma \vdash_{\cap CC} e_1\ (blame_{T_2}\ l) : T_1}{e_1\ (blame_{T_2}\ l) \longrightarrow_{\cap CC} blame_{T_1}\ l} \text{ E-PushBlame2}$$

$$\frac{\vdash_{\cap IC} c_1 : T_1 \ldots \vdash_{\cap IC} c_n : T_n}{blame_T\ l : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} blame_{T_1 \cap \ldots \cap T_n}\ l} \text{ E-PushBlameCast}$$

Evaluate expressions

$$\frac{e_1 \longrightarrow_{\cap CC} e_1'}{e_1\ e_2 \longrightarrow_{\cap CC}\ e_1'\ e_2} \text{ E-App1} \qquad \frac{e_2 \longrightarrow_{\cap CC} e_2'}{v_1\ e_2 \longrightarrow_{\cap CC}\ v_1\ e_2'} \text{ E-App2}$$

$$\frac{}{(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v \longrightarrow_{\cap CC}\ [x \mapsto v]e} \text{ E-AppAbs}$$

$$\frac{e \longrightarrow_{\cap CC} e'}{e : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} e' : c_1 \cap \ldots \cap c_n} \text{ E-Evaluate}$$

Simulate casts on data types

$$\frac{\begin{array}{c} \textit{is value}\ (v_1 : cv_1 \cap \ldots \cap cv_n) \qquad \exists i \in 1..n\ .\ \textit{isArrowCompatible}(cv_i) \\ ((c_{11}, c_{12}, c_1^s), \ldots, (c_{m1}, c_{m2}, c_m^s)) = \textit{simulateArrow}(cv_1, \ldots, cv_n) \end{array}}{\begin{array}{c} (v_1 : cv_1 \cap \ldots \cap cv_n)\ v_2 \longrightarrow_{\cap CC} \\ (v_1 : c_1^s \cap \ldots \cap c_m^s)\ (v_2 : c_{11} \cap \ldots \cap c_{m1}) : c_{12} \cap \ldots \cap c_{m2} \end{array}} \text{ E-SimulateArrow}$$

Merge casts

$$\frac{\begin{array}{c} \textit{is value}\ (v : cv_1 \cap \ldots \cap cv_n) \\ v : c_1'' \cap \ldots \cap c_j'' = \textit{mergeCasts}(v : cv_1 \cap \ldots \cap cv_n : c_1' \cap \ldots \cap c_m') \end{array}}{v : cv_1 \cap \ldots \cap cv_n : c_1' \cap \ldots \cap c_m' \longrightarrow_{\cap CC} v : c_1'' \cap \ldots \cap c_j''} \text{ E-MergeCasts}$$

Evaluate intersection casts

$$\frac{\neg(\forall i \in 1..n\ .\ \textit{is cast value}\ c_i) \qquad c_1 \longrightarrow_{\cap IC} cv_1\ \ \ldots\ \ c_n \longrightarrow_{\cap IC} cv_n}{v : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} v : cv_1 \cap \ldots \cap cv_n} \text{ E-EvaluateCasts}$$

Transition from cast values to values

$$\frac{}{v : \textit{blame}\ I_1\ F_1\ l_1\ ^{cl_1} \cap \ldots \cap \textit{blame}\ I_n\ F_n\ l_n\ ^{cl_n} \longrightarrow_{\cap CC} \textit{blame}_{(F_1 \cap \ldots \cap F_n)}\ l_1} \text{ E-PropagateBlame}$$

$$\frac{}{v : \varnothing\ T_1\ ^{cl_1} \cap \ldots \cap \varnothing\ T_n\ ^{cl_n} \longrightarrow_{\cap CC} v} \text{ E-RemoveEmpty}$$

Figure 6: Cast Calculus Operational Semantics ($\longrightarrow_{\cap CC}$)

$$\boxed{\langle c \rangle^{cl} = \text{c}}$$

$$\langle c : T_1 \Rightarrow^l T_2 \ ^{cl} \rangle^{cl'} = \langle c \rangle^{cl'} : T_1 \Rightarrow^l T_2 \ ^{cl'}$$

$$\langle \mathit{blame} \ T_I \ T_F \ l \ ^{cl'} \rangle^{cl} = \mathit{blame} \ T_I \ T_F \ l \ ^{cl}$$

$$\langle \varnothing \ T \ ^{cl'} \rangle^{cl} = \varnothing \ T \ ^{cl}$$

$$\boxed{\text{isArrowCompatible(c)} = \text{Bool}}$$

$$\mathit{isArrowCompatible}(c : T_{11} \to T_{12} \Rightarrow^l T_{21} \to T_{22} \ ^{cl}) = \mathit{isArrowCompatible}(c)$$

$$\mathit{isArrowCompatible}(\varnothing \ (T_1 \to T_2) \ ^{cl}) = \mathit{True}$$

$$\boxed{\mathit{separateIntersectionCast}(c) = \text{(c, c)}}$$

$$\mathit{separateIntersectionCast}(c : T_1 \Rightarrow^l T_2 \ ^{cl}) = (\varnothing \ T_1 \ ^{cl} : T_1 \Rightarrow^l T_2 \ ^{cl}, c)$$

$$\mathit{separateIntersectionCast}(\varnothing \ T \ ^{cl}) = (\varnothing \ T \ ^{cl}, \varnothing \ T \ ^{cl})$$

$$\boxed{\mathit{breakdownArrowType}(c) = \text{(c, c)}}$$

$$\begin{gathered}\mathit{breakdownArrowType}(\varnothing \ T_{11} \to T_{12} \ ^{cl} : T_{11} \to T_{12} \Rightarrow^l T_{21} \to T_{22} \ ^{cl}) = \\ (\varnothing \ T_{21} \ ^{cl} : T_{21} \Rightarrow^l T_{11} \ ^{cl}, \varnothing \ T_{12} \ ^{cl} : T_{12} \Rightarrow^l T_{22} \ ^{cl})\end{gathered}$$

$$\mathit{breakdownArrowType}(\varnothing \ T_1 \to T_2 \ ^{cl}) = (\varnothing \ T_1 \ ^{cl}, \varnothing \ T_2 \ ^{cl} \ )$$

$$\boxed{\text{simulateArrow}(c_1, \ldots, c_n) = ((c_{11}, c_{12}, c_1^s), \ldots, (c_{m1}, c_{m2}, c_m^s))}$$

$$\frac{\begin{gathered}(c'_1, \ldots, c'_m) = \mathit{filter} \ \mathit{isArrowCompatible} \ (c_1, \ldots, c_n) \\ ((c_1^f, c_1^s), \ldots, (c_m^f, c_m^s)) = \mathit{map} \ \mathit{separateIntersectionCast} \ (\langle c'_1 \rangle^0, \ldots, \langle c'_m \rangle^0) \\ ((c_{11}, c_{12}), \ldots, (c_{m1}, c_{m2})) = \mathit{map} \ \mathit{breakdownArrowType} \ (\langle c_1^f \rangle^1, \ldots, \langle c_m^f \rangle^m)\end{gathered}}{\mathit{simulateArrow}(c_1, \ldots, c_n) = ((c_{11}, c_{12}, c_1^s), \ldots, (c_{m1}, c_{m2}, c_m^s))}$$

Figure 7: Definitions for auxiliary semantic functions

$\boxed{\text{getCastLabel(c) = cl}}$

$$getCastLabel(c : T_1 \Rightarrow^l T_2 \ ^{cl}) = cl$$

$$getCastLabel(blame\ T_I\ T_F\ l\ ^{cl}) = cl$$

$$getCastLabel(\varnothing\ T\ ^{cl}) = cl$$

$\boxed{\text{sameCastLabel(c, c) = Bool}}$

$$sameCastLabel(c_1, c_2) = getCastLabel(c_1) == 0$$

$$sameCastLabel(c_1, c_2) = getCastLabel(c_2) == 0$$

$$sameCastLabel(c_1, c_2) = getCastLabel(c_1) == getCastLabel(c_2)$$

$\boxed{\text{joinCasts(c, c) = c}}$

$$joinCasts(c : T_1 \Rightarrow^l T_2 \ ^{cl}, c') = joinCasts(c, c') : T_1 \Rightarrow^l T_2 \ ^{cl}$$

$$joinCasts(blame\ T_I\ T_F\ l\ ^{cl}, c) = blame\ T_I\ T_F\ l\ ^{cl}$$

$$getCastLabel(\varnothing\ T\ ^{cl}, c) = \langle \mathrm{c} \rangle^{cl}$$

$\boxed{mergeCasts(e) = e}$

$$\frac{\begin{array}{c}(c'_1, \ldots, c'_o) = [joinCast\ y\ x \mid x \leftarrow (c_{11}, \ldots, c_{1m}),\ y \leftarrow (c_{21}, \ldots, c_{2n}),\\ sameCastLabel\ y\ x\ \&\&\ initialType(y) == finalType(x)]\end{array}}{mergeCasts(e : c_{11} \cap \ldots \cap c_{1m} : c_{21} \cap \ldots \cap c_{2n}) = e : c'_1 \cap \ldots \cap c'_o}$$

Figure 8: Definitions for auxiliary semantic functions

$\boxed{\vdash_{\cap IC}\ c : T}$ Typing

$$\frac{\vdash_{\cap IC}\ c : T_1 \qquad T_1 \sim T_2}{\vdash_{\cap IC}\ (c : T_1 \Rightarrow^{l} T_2\ ^{cl}) : T_2}\ \text{T-SingleIC} \qquad \frac{}{\vdash_{\cap IC}\ \mathit{blame}\ T_I\ T_F\ l\ ^{cl} : T_F}\ \text{T-BlameIC}$$

$$\frac{}{\vdash_{\cap IC}\ \varnothing\ T\ ^{cl} : T}\ \text{T-EmptyIC}$$

Figure 9: Intersection Casts Type System ($\vdash_{\cap IC}$)

$\boxed{c \longrightarrow_{\cap IC} c}$ Evaluation

Push blame to top level

$$\frac{}{\mathit{blame}\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2} \longrightarrow_{\cap IC} \mathit{blame}\ T_I\ T_2\ l_1\ ^{cl_1}}\ \text{E-PushBlameIC}$$

Evaluate inside casts

$$\frac{\neg(\mathit{is\ cast\ value}\ c) \qquad c \longrightarrow_{\cap IC} c'}{c : T_1 \Rightarrow^{l} T_2\ ^{cl} \longrightarrow_{\cap IC} c' : T_1 \Rightarrow^{l} T_2\ ^{cl}}\ \text{E-EvaluateIC}$$

Detect success or failure of casts

$$\frac{\mathit{is\ cast\ value}\ 1\ c \vee \mathit{is\ empty\ cast}\ c}{c : T \Rightarrow^{l} T\ ^{cl} \longrightarrow_{\cap IC} c}\ \text{E-IdentityIC}$$

$$\frac{\mathit{is\ cast\ value}\ 1\ c \vee \mathit{is\ empty\ cast}\ c}{c : G \Rightarrow^{l_1} \mathit{Dyn}\ ^{cl_1} : \mathit{Dyn} \Rightarrow^{l_2} G\ ^{cl_2} \longrightarrow_{\cap IC} c}\ \text{E-SucceedIC}$$

$$\frac{\mathit{is\ cast\ value}\ 1\ c \vee \mathit{is\ empty\ cast}\ c \qquad \neg(\mathit{same\ ground}\ G_1\ G_2) \qquad \mathit{initialType}(c) = T_I}{c : G_1 \Rightarrow^{l_1} \mathit{Dyn}\ ^{cl_1} : \mathit{Dyn} \Rightarrow^{l_2} G_2\ ^{cl_2} \longrightarrow_{\cap IC} \mathit{blame}\ T_I\ G_2\ l_2\ ^{cl_1}}\ \text{E-FailIC}$$

Mediate the transition between the two disciplines

$$\frac{\mathit{is\ cast\ value}\ 1\ c \vee \mathit{is\ empty\ cast}\ c \qquad G\ \mathit{is\ ground\ type\ of}\ T \qquad \neg(\mathit{ground}\ T)}{c : T \Rightarrow^{l} \mathit{Dyn}\ ^{cl} \longrightarrow_{\cap IC} c : T \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} \mathit{Dyn}\ ^{cl}}\ \text{E-GroundIC}$$

$$\frac{\mathit{is\ cast\ value}\ 1\ c \vee \mathit{is\ empty\ cast}\ c \qquad G\ \mathit{is\ ground\ type\ of}\ T \qquad \neg(\mathit{ground}\ T)}{c : \mathit{Dyn} \Rightarrow^{l} T\ ^{cl} \longrightarrow_{\cap IC} c : \mathit{Dyn} \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} T\ ^{cl}}\ \text{E-ExpandIC}$$

Figure 10: Intersection Casts Operational Semantics ($\longrightarrow_{\cap IC}$)

2 Proofs

Lemma 1 (Consistency reduces to equality when comparing static types)**.** *If T_1 and T_2 are static types then $T_1 = T_2 \iff T_1 \sim T_2$.*

Proof. We proceed by structural induction on T_1.

Base cases:

- $T_1 = Int$.
  - If $Int = Int$ then, by the definition of $\sim$, $Int \sim Int$.
  - If $Int \sim Int$, then $Int = Int$.
- $T_1 = Bool$.
  - If $Bool = Bool$ then, by the definition of $\sim$, $Bool \sim Bool$.
  - If $Bool \sim Bool$, then $Bool = Bool$.

Induction step:

- $T_1 = T_{11} \to T_{12}$.
  - If $T_{11} \to T_{12} = T_{21} \to T_{22}$, for some T_{21} and T_{22}, then $T_{11} = T_{21}$ and $T_{12} = T_{22}$. By the induction hypothesis, $T_{11} \sim T_{21}$ and $T_{12} \sim T_{22}$. Therefore, by the definition of $\sim$, $T_{11} \to T_{12} \sim T_{21} \to T_{22}$.
  - If $T_{11} \to T_{12} \sim T_2$, then by the definition of $\sim$, $T_2 = T_{21} \to T_{22}$ and $T_{11} \sim T_{21}$ and $T_{12} \sim T_{22}$. By the induction hypothesis, $T_{11} = T_{21}$ and $T_{12} = T_{22}$. Therefore, $T_{11} \to T_{12} = T_{21} \to T_{22}$.
- $T_1 = T_{11} \cap \ldots \cap T_{1n}$.
  - If $T_{11} \cap \ldots \cap T_{1n} = T_2$, then $\exists T_{21} \ldots T_{2n} \,.\, T_2 = T_{21} \cap \ldots \cap T_{2n}$ and $T_{11} = T_{21}$ and ... and $T_{1n} = T_{2n}$. By the induction hypothesis, $T_{11} \sim T_{21}$ and ... and $T_{1n} \sim T_{2n}$. Therefore, by the definition of $\sim$, $T_{11} \cap \ldots \cap T_{1n} \sim T_{21} \cap \ldots \cap T_{2n}$.
  - If $T_{11} \cap \ldots \cap T_{1n} \sim T_2$, then either:
    - $\exists T_{21} \ldots T_{2n} \,.\, T_2 = T_{21} \cap \ldots \cap T_{2n}$ and $T_{11} \sim T_{21}$ and ... and $T_{1n} \sim T_{2n}$. By the induction hypothesis, $T_{11} = T_{21}$ and ... and $T_{1n} = T_{2n}$. Therefore, $T_{11} \cap \ldots \cap T_{1n} = T_{21} \cap \ldots \cap T_{2n}$.
    - $T_{11} \sim T_2$ and ... and $T_{1n} \sim T_2$. By the induction hypothesis, $T_{11} = T_2$ and ... and $T_{1n} = T_2$. As $T_2 \cap \ldots \cap T_2 = T_2$, then $T_{11} \cap \ldots \cap T_{1n} = T_2$.

□

Theorem 1 (Conservative Extension)**.** *Depends on Lemma 1. If e is fully static and T is a static type, then $\Gamma \vdash_{\cap S} e : T \iff \Gamma \vdash_{\cap G} e : T$.*

Proof. We proceed by induction on the length of the derivation tree of $\vdash_{\cap S}$ and $\vdash_{\cap G}$ for the right and left direction of the implication, respectively.

Base cases:

- Rule T-Var.
  - If $\Gamma \vdash_{\cap S} x : T$, then $x : T \in \Gamma$. Therefore, $\Gamma \vdash_{\cap G} x : T$.
  - If $\Gamma \vdash_{\cap G} x : T$, then $x : T \in \Gamma$. Therefore, $\Gamma \vdash_{\cap S} e : T$.
- Rule T-Int.
  - If $\Gamma \vdash_{\cap S} n : Int$, then $\Gamma \vdash_{\cap G} n : Int$.
  - If $\Gamma \vdash_{\cap G} n : Int$, then $\Gamma \vdash_{\cap S} n : Int$.
- Rule T-True.
  - If $\Gamma \vdash_{\cap S} true : Bool$, then $\Gamma \vdash_{\cap G} true : Bool$.
  - If $\Gamma \vdash_{\cap G} true : Bool$, then $\Gamma \vdash_{\cap S} true : Bool$.
- Rule T-False.
  - If $\Gamma \vdash_{\cap S} false : Bool$, then $\Gamma \vdash_{\cap G} false : Bool$.
  - If $\Gamma \vdash_{\cap G} false : Bool$, then $\Gamma \vdash_{\cap S} false : Bool$.

Induction step:

- Rule T-Abs.
  - If $\Gamma \vdash_{\cap S} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_1 \cap \ldots \cap T_n \to T$, then $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap S} e : T$. By the induction hypothesis, $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap G} e : T$. Therefore, $\Gamma \vdash_{\cap G} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_1 \cap \ldots \cap T_n \to T$.
  - If $\Gamma \vdash_{\cap G} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_1 \cap \ldots \cap T_n \to T$, then $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap G} e : T$. By the induction hypothesis, $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap S} e : T$. Therefore, $\Gamma \vdash_{\cap S} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_1 \cap \ldots \cap T_n \to T$.
- Rule T-Abs'.
  - If $\Gamma \vdash_{\cap S} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_i \to T$, then $\Gamma, x : T_i \vdash_{\cap S} e : T$. By the induction hypothesis, $\Gamma, x : T_i \vdash_{\cap G} e : T$. Therefore, $\Gamma \vdash_{\cap G} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_i \to T$.
  - If $\Gamma \vdash_{\cap G} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_i \to T$, then $\Gamma, x : T_i \vdash_{\cap G} e : T$. By the induction hypothesis, $\Gamma, x : T_i \vdash_{\cap S} e : T$. Therefore, $\Gamma \vdash_{\cap S} \lambda x \,.\, T_1 \cap \ldots \cap T_n \,.\, e : T_i \to T$.
- Rule T-App.
  - If $\Gamma \vdash_{\cap S} e_1\ e_2 : T$ then $\Gamma \vdash_{\cap S} e_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap S} e_2 : T_1 \cap \ldots \cap T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap G} e_2 : T_1 \cap \ldots \cap T_n$. By the definition of $\rhd$, $T_1 \cap \ldots \cap T_n \to T \rhd T_1 \cap \ldots \cap T_n \to T$. By the definition of $\sim$, $T_1 \cap \ldots \cap T_n \sim T_1 \cap \ldots \cap T_n$. Therefore, $\Gamma \vdash_{\cap G} e_1\ e_2 : T$.
  - If $\Gamma \vdash_{\cap G} e_1\ e_2 : T$ then $\Gamma \vdash_{\cap G} e_1 : PM$, $PM \rhd T_1 \cap \ldots \cap T_n \to T$, $\Gamma \vdash_{\cap G} e_2 : T_1' \cap \ldots \cap T_n'$ and $T_1' \cap \ldots \cap T_n' \sim T_1 \cap \ldots \cap T_n$. By the definition of $\rhd$, $PM = T_1 \cap \ldots \cap T_n \to T$, therefore $\Gamma \vdash_{\cap G} e_1 : T_1 \cap \ldots \cap T_n \to T$. By Lemma 1, $T_1' \cap \ldots \cap T_n' = T_1 \cap \ldots \cap T_n$, and therefore $\Gamma \vdash_{\cap G} e_2 : T_1 \cap \ldots \cap T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap S} e_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap S} e_2 : T_1 \cap \ldots \cap T_n$. Therefore, $\Gamma \vdash_{\cap S} e_1\ e_2 : T$.
- Rule T-Gen.

- If $\Gamma \vdash_{\cap S} e : T_1 \cap \ldots \cap T_n$ then $\Gamma \vdash_{\cap S} e : T_1$ and ... and $\Gamma \vdash_{\cap S} e : T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e : T_1$ and ... and $\Gamma \vdash_{\cap G} e : T_n$. Therefore, $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$.
- If $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$ then $\Gamma \vdash_{\cap G} e : T_1$ and ... and $\Gamma \vdash_{\cap G} e : T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap S} e : T_1$ and ... and $\Gamma \vdash_{\cap S} e : T_n$. Therefore $\Gamma \vdash_{\cap S} e : T_1 \cap \ldots \cap T_n$.

- Rule T-Inst.
  - If $\Gamma \vdash_{\cap S} e : T_i$ then $\Gamma \vdash_{\cap S} e : T_1 \cap \ldots \cap T_n$, such that $T_i \in \{T_1, ..., T_n\}$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$. As $T_i \in \{T_1, ..., T_n\}$, then $\Gamma \vdash_{\cap G} e : T_i$.
  - If $\Gamma \vdash_{\cap G} e : T_i$ then $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$, such that $T_i \in \{T_1, ..., T_n\}$. By the induction hypothesis, $\Gamma \vdash_{\cap S} e : T_1 \cap \ldots \cap T_n$. As $T_i \in \{T_1, ..., T_n\}$, then $\Gamma \vdash_{\cap S} e : T_i$.

□

Theorem 2 (Monotonicity w.r.t. precision)**.** *If $\Gamma \vdash_{\cap G} e : T$ and $e' \sqsubseteq e$ then $\Gamma \vdash_{\cap G} e' : T'$ and $T' \sqsubseteq T$.*

Proof. We proceed by induction on the length of the derivation tree of $\Gamma \vdash_{\cap G} e : T$.

Base cases:

- Rule T-Var. If $\Gamma \vdash_{\cap G} x : T$ and $x \sqsubseteq x$, then $\Gamma \vdash_{\cap G} x : T$ and $T \sqsubseteq T$.
- Rule T-Int. If $\Gamma \vdash_{\cap G} n : Int$ and $n \sqsubseteq n$, then $\Gamma \vdash_{\cap G} n : Int$ and $Int \sqsubseteq Int$.
- Rule T-True. If $\Gamma \vdash_{\cap G} true : Bool$ and $true \sqsubseteq true$, then $\Gamma \vdash_{\cap G} true : Bool$ and $Bool \sqsubseteq Bool$.
- Rule T-False. If $\Gamma \vdash_{\cap G} false : Bool$ and $false \sqsubseteq false$, then $\Gamma \vdash_{\cap G} false : Bool$ and $Bool \sqsubseteq Bool$.

Induction step:

- Rule T-Abs. If $\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n \,.\, e : T_1 \cap \ldots \cap T_n \rightarrow T$ and $\lambda x : T'_1 \cap \ldots \cap T'_n \,.\, e' \sqsubseteq \lambda x : T_1 \cap \ldots \cap T_n \,.\, e$, then by rule T-Abs, $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap G} e : T$, and by the definition of $\sqsubseteq$, $T'_1 \cap \ldots \cap T'_n \sqsubseteq T_1 \cap \ldots \cap T_n$ and $e' \sqsubseteq e$. By the induction hypothesis, $\Gamma, x : T'_1 \cap \ldots \cap T'_n \vdash_{\cap G} e' : T'$ and $T' \sqsubseteq T$. By rule T-Abs, $\Gamma \vdash_{\cap G} \lambda x : T'_1 \cap \ldots \cap T'_n \,.\, e' : T'_1 \cap \ldots \cap T'_n \rightarrow T'$, and by the definition of $\sqsubseteq$, $T'_1 \cap \ldots \cap T'_n \rightarrow T' \sqsubseteq T_1 \cap \ldots \cap T_n \rightarrow T$.
- Rule T-Abs'. If $\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n \,.\, e : T_i \rightarrow T$ and $\lambda x : T'_1 \cap \ldots \cap T'_n \,.\, e' \sqsubseteq \lambda x : T_1 \cap \ldots \cap T_n \,.\, e$, then by rule T-Abs', $\Gamma, x : T_i \vdash_{\cap G} e : T$, and by the definition of $\sqsubseteq$, $T'_1 \cap \ldots \cap T'_n \sqsubseteq T_1 \cap \ldots \cap T_n$ and $e' \sqsubseteq e$. By the induction hypothesis, $\Gamma, x : T'_i \vdash_{\cap G} e' : T'$ and $T' \sqsubseteq T$. By rule T-Abs', $\Gamma \vdash_{\cap G} \lambda x : T'_1 \cap \ldots \cap T'_n \,.\, e' : T'_i \rightarrow T'$, and by the definition of $\sqsubseteq$, $T'_i \rightarrow T' \sqsubseteq T_i \rightarrow T$.
- Rule T-App. If $\Gamma \vdash_{\cap G} e_1\ e_2 : T$ and $e'_1\ e'_2 \sqsubseteq e_1\ e_2$ then by rule T-App, $\Gamma \vdash_{\cap G} e_1 : PM$, $PM \rhd T_{11} \cap \ldots \cap T_{1n} \rightarrow T$, $\Gamma \vdash_{\cap G} e_2 : T_{21} \cap \ldots \cap T_{2n}$, and $T_{21} \cap \ldots \cap T_{2n} \sim T_{11} \cap \ldots \cap T_{1n}$, and by the definition of $\sqsubseteq$, $e'_1 \sqsubseteq e_1$ and $e'_2 \sqsubseteq e_2$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e'_1 : PM'$ *and* $PM' \sqsubseteq PM$ *and* $PM' \rhd T'_{11} \cap \ldots \cap T'_{1n} \rightarrow T'$ and $\Gamma \vdash_{\cap G} e'_2 : T'_{21} \cap \ldots \cap T'_{2n}$ *and* $T'_{21} \cap \ldots \cap T'_{2n} \sqsubseteq T_{21} \cap \ldots \cap T_{2n}$ *and* $T'_{21} \cap \ldots \cap T'_{2n} \sim T'_{11} \cap \ldots \cap T'_{1n}$. By the definition of $\sqsubseteq$ and $\rhd$, $T'_{11} \cap \ldots \cap T'_{1n} \rightarrow T' \sqsubseteq T_{11} \cap \ldots \cap T_{1n} \rightarrow T$, and therefore, $T' \sqsubseteq T$. As $\Gamma \vdash_{\cap G} e'_1\ e'_2 : T'$, it is proved.

- Rule T-Gen. If $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$ and $e' \sqsubseteq e$, then by rule T-Gen, $\Gamma \vdash_{\cap G} e : T_1$ and ... and $\Gamma \vdash_{\cap G} e : T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e' : T_1'$ *and* $T_1' \sqsubseteq T_1$ and ... and $\Gamma \vdash_{\cap G} e' : T_n'$ *and* $T_n' \sqsubseteq T_n$. Then by rule T-Gen, $\Gamma \vdash_{\cap G} e' : T_1' \cap \ldots \cap T_n'$ and by the definition of $\sqsubseteq$, $T_1' \cap \ldots \cap T_n' \sqsubseteq T_1 \cap \ldots \cap T_n$.

- Rule T-Inst. If $\Gamma \vdash_{\cap G} e : T_i$ and $e' \sqsubseteq e$, then by rule T-Inst, $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$ such that $T_i \in \{T_1, ..., T_n\}$. By the induction hypothesis, $\Gamma \vdash_{\cap G} e' : T_1' \cap \ldots \cap T_n'$ and $T_1' \cap \ldots \cap T_n' \sqsubseteq T_1 \cap \ldots \cap T_n$. Therefore, by rule T-Inst, $\Gamma \vdash_{\cap G} e' : T_i'$ and by the definition of $\sqsubseteq$, $T_i' \sqsubseteq T_i$.

$\square$

Theorem 3 (Type preservation of cast insertion)**.** *If* $\Gamma \vdash_{\cap G} e : T$ *then* $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T$ *and* $\Gamma \vdash_{\cap CC} e' : T$.

Proof. We proceed by induction on the length of the derivation tree of $\Gamma \vdash_{\cap G} e : T$.

Base cases:

- Rule T-Var. If $\Gamma \vdash_{\cap G} x : T$, then by rule T-Var, $x : T \in \Gamma$. By rule C-Var, $\Gamma \vdash_{\cap CC} x \rightsquigarrow x : T$ and by rule T-Var, $\Gamma \vdash_{\cap CC} x : T$.

- Rule T-Int. As $\Gamma \vdash_{\cap G} n : Int$, then by rule C-Int, $\Gamma \vdash_{\cap CC} n \rightsquigarrow n : Int$ and by rule T-Int, $\Gamma \vdash_{\cap CC} n : Int$.

- Rule T-True. As $\Gamma \vdash_{\cap G} true : Bool$, then by rule C-True, $\Gamma \vdash_{\cap CC} true \rightsquigarrow true : Bool$ and by rule T-True, $\Gamma \vdash_{\cap CC} true : Bool$.

- Rule T-False. As $\Gamma \vdash_{\cap G} false : Bool$, then by rule C-False, $\Gamma \vdash_{\cap CC} false \rightsquigarrow false : Bool$ and by rule T-False, $\Gamma \vdash_{\cap CC} false : Bool$, it is proved.

Induction step:

- Rule T-Abs. If $\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e : T_1 \cap \ldots \cap T_n \rightarrow T$ then by rule T-Abs, $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap G} e : T$. By the induction hypothesis, $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap CC} e \rightsquigarrow e' : T$ and $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap CC} e' : T$. By rule C-Abs, $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e \rightsquigarrow \lambda x : T_1 \cap \ldots \cap T_n \ . \ e' : T_1 \cap \ldots \cap T_n \rightarrow T$ and by rule T-Abs, $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e' : T_1 \cap \ldots \cap T_n \rightarrow T$.

- Rule T-Abs'. If $\Gamma \vdash_{\cap G} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e : T_i \rightarrow T$ then by rule T-Abs', $\Gamma, x : T_i \vdash_{\cap G} e : T$. By the induction hypothesis, $\Gamma, x : T_i \vdash_{\cap CC} e \rightsquigarrow e' : T$ and $\Gamma, x : T_i \vdash_{\cap CC} e' : T$. By rule C-Abs', $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e \rightsquigarrow \lambda x : T_1 \cap \ldots \cap T_n \ . \ e' : T_i \rightarrow T$ and by rule T-Abs', $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n \ . \ e' : T_i \rightarrow T$.

- Rule T-App. If $\Gamma \vdash_{\cap G} e_1 \ e_2 : T$ then by rule T-App, $\Gamma \vdash_{\cap G} e_1 : PM$, $PM \rhd T_1 \cap \ldots \cap T_n \rightarrow T$, $\Gamma \vdash_{\cap G} e_2 : T_1' \cap \ldots \cap T_n'$ and $T_1' \cap \ldots \cap T_n' \sim T_1 \cap \ldots \cap T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : PM$ and $\Gamma \vdash_{\cap CC} e_1' : PM$, and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e_2' : T_1' \cap \ldots \cap T_n'$ and $\Gamma \vdash_{\cap CC} e_2' : T_1' \cap \ldots \cap T_n'$. Therefore, by rule C-App, $\Gamma \vdash_{\cap CC} e_1 \ e_2 \rightsquigarrow e_1'' \ e_2'' : T$. By the definition of *instances* and S, $S, \ e \hookrightarrow e$, by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} e_1'' : T_1 \rightarrow T \cap \ldots \cap T_n \rightarrow T$ and $\Gamma \vdash_{\cap CC} e_2'' : T_1 \cap \ldots \cap T_n$. By rule T-App', $\Gamma \vdash_{\cap CC} e_1'' \ e_2'' : T \cap \ldots \cap T$ and then by the properties of intersection types (modulo repetitions), $\Gamma \vdash_{\cap CC} e_1'' \ e_2'' : T$.

- Rule T-Gen. If $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$ then by rule T-Gen, $\Gamma \vdash_{\cap G} e : T_1$ and ... and $\Gamma \vdash_{\cap G} e : T_n$. By the induction hypothesis, $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1$ and ... and $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_n$, and $\Gamma \vdash_{\cap CC} e' : T_1$ and ... and $\Gamma \vdash_{\cap CC} e' : T_n$. By rule C-Gen, $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1 \cap \ldots \cap T_n$ and by rule T-Gen, $\Gamma \vdash_{\cap CC} e' : T_1 \cap \ldots \cap T_n$.

- Rule T-Inst. If $\Gamma \vdash_{\cap G} e : T_i$ then by rule T-Inst, $\Gamma \vdash_{\cap G} e : T_1 \cap \ldots \cap T_n$, such that $T_i \in \{T_1, \ldots, T_n\}$. By the induction hypothesis, $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_1 \cap \ldots \cap T_n$ and $\Gamma \vdash_{\cap CC} e' : T_1 \cap \ldots \cap T_n$. By rule C-Inst, $\Gamma \vdash_{\cap CC} e \rightsquigarrow e' : T_i$ and by rule T-Inst, $\Gamma \vdash_{\cap CC} e' : T_i$.

□

Theorem 4 (Monotonicity w.r.t precision of cast insertion)**.** *If $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : T_1$ and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e_2' : T_2$ and $e_1 \sqsubseteq e_2$ then $e_1' \sqsubseteq e_2'$ and $T_1 \sqsubseteq T_2$.*

Proof. We proceed by induction on the length of the derivation tree of $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : T$. Base cases:

- Rule C-Var. If $\Gamma \vdash_{\cap CC} x \rightsquigarrow x : T$ and $\Gamma \vdash_{\cap CC} x \rightsquigarrow x : T$, and $x \sqsubseteq x$, then $x \sqsubseteq x$ and $T \sqsubseteq T$.

- Rule C-Int. If $\Gamma \vdash_{\cap CC} n \rightsquigarrow n : Int$, $\Gamma \vdash_{\cap CC} n \rightsquigarrow n : Int$ and $n \sqsubseteq n$, then $n \sqsubseteq n$ and $Int \sqsubseteq Int$.

- Rule C-True. If $\Gamma \vdash_{\cap CC} true \rightsquigarrow true : Bool$, $\Gamma \vdash_{\cap CC} true \rightsquigarrow true : Bool$ and $true \sqsubseteq true$, then $true \sqsubseteq true$ and $Bool \sqsubseteq Bool$.

- Rule C-False. If $\Gamma \vdash_{\cap CC} false \rightsquigarrow false : Bool$, $\Gamma \vdash_{\cap CC} false \rightsquigarrow false : Bool$ and $false \sqsubseteq false$, then $false \sqsubseteq false$ and $Bool \sqsubseteq Bool$.

Induction step:

- Rule C-Abs. If $\Gamma \vdash_{\cap CC} \lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1 \rightsquigarrow \lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1' : T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1$ and $\Gamma \vdash_{\cap CC} \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2 \rightsquigarrow \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2' : T_{21} \cap \ldots \cap T_{2n} \rightarrow T_2$ and $\lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1 \sqsubseteq \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2$ then by rule C-Abs, $\Gamma, x : T_{11} \cap \ldots \cap T_{1n} \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : T_1$ and $\Gamma, x : T_{21} \cap \ldots \cap T_{2n} \vdash_{\cap CC} e_2 \rightsquigarrow e_2' : T_2$ and by the definition of $\sqsubseteq$, $T_{11} \cap \ldots \cap T_{1n} \sqsubseteq T_{21} \cap \ldots \cap T_{2n}$ and $e_1 \sqsubseteq e_2$. By the induction hypothesis, $e_1' \sqsubseteq e_2'$ and $T_1 \sqsubseteq T_2$. Therefore, by the definition of $\sqsubseteq$, $\lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1' \sqsubseteq \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2'$ and $T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1 \sqsubseteq T_{21} \cap \ldots \cap T_{2n} \rightarrow T_2$.

- Rule C-Abs'. If $\Gamma \vdash_{\cap CC} \lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1 \rightsquigarrow \lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1' : T_{1i} \rightarrow T_1$, such that $T_{1i} \in \{T_{11}, \ldots, T_{1n}\}$, and $\Gamma \vdash_{\cap CC} \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2 \rightsquigarrow \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2' : T_{2i} \rightarrow T_2$, such that $T_{2i} \in \{T_{21}, \ldots, T_{2n}\}$, and $\lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1 \sqsubseteq \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2$ then by the definition of C-Abs', $\Gamma, x : T_{1i} \vdash_{\cap CC} e_1 \rightsquigarrow e_1' : T_1$ and $\Gamma, x : T_{2i} \vdash_{\cap CC} e_2 \rightsquigarrow e_2' : T_2$ and by the definition of $\sqsubseteq$, $T_{11} \cap \ldots \cap T_{1n} \sqsubseteq T_{21} \cap \ldots \cap T_{2n}$ and $e_1 \sqsubseteq e_2$ and therefore $T_{1i} \sqsubseteq T_{2i}$. By the induction hypothesis, $e_1' \sqsubseteq e_2'$ and $T_1 \sqsubseteq T_2$. Therefore, by the definition of $\sqsubseteq$, $\lambda x : T_{11} \cap \ldots \cap T_{1n} \, . \, e_1' \sqsubseteq \lambda x : T_{21} \cap \ldots \cap T_{2n} \, . \, e_2'$ and $T_{1i} \rightarrow T_1 \sqsubseteq T_{2i} \rightarrow T_2$.

- Rule C-App. If $\Gamma \vdash_{\cap CC} e_{11} \; e_{12} \rightsquigarrow e_{11}'' \; e_{12}'' : T_1$ and $\Gamma \vdash_{\cap CC} e_{21} \; e_{22} \rightsquigarrow e_{21}'' \; e_{22}'' : T_2$ and $e_{11} \; e_{12} \sqsubseteq e_{21} \; e_{22}$ then by rule C-App, $\Gamma \vdash_{\cap CC} e_{11} \rightsquigarrow e_{11}' : PM_1$ and $PM_1 \triangleright T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1$ and $\Gamma \vdash_{\cap CC} e_{12} \rightsquigarrow e_{12}' : T_{11}' \cap \ldots \cap T_{1n}'$ and $T_{11}' \cap \ldots \cap T_{1n}' \sim T_{11} \cap \ldots \cap T_{1n}$ and $instances(PM_1) = S_{11}$ and $instances(T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1) = S_{12}$ and $instances(T_{11}' \cap \ldots \cap T_{1n}') = S_{13}$ and $instances(T_{11} \cap \ldots \cap T_{1n}) = S_{14}$ and S_{11}, S_{12}, $e_{11}' \hookrightarrow e_{11}''$ and S_{13}, S_{14}, $e_{12}' \hookrightarrow e_{12}''$ and $\Gamma \vdash_{\cap CC} e_{21} \rightsquigarrow e_{21}' : PM_2$ and $PM_2 \triangleright T_{21} \cap \ldots \cap T_{2n} \rightarrow T_2$ and $\Gamma \vdash_{\cap CC} e_{22} \rightsquigarrow e_{22}' : T_{21}' \cap \ldots \cap T_{2n}'$ and $T_{21}' \cap \ldots \cap T_{2n}' \sim T_{21} \cap \ldots \cap T_{2n}$ and $instances(PM_2) = S_{21}$ and $instances(T_{21} \cap \ldots \cap T_{2n} \rightarrow$

$T_2) = S_{22}$ and $instances(T'_{21} \cap \ldots \cap T'_{2n}) = S_{23}$ and $instances(T_{21} \cap \ldots \cap T_{2n}) = S_{24}$ and $S_{21},\ S_{22},\ e'_{21} \hookrightarrow e''_{21}$ and $S_{23},\ S_{24},\ e'_{22} \hookrightarrow e''_{22}$. As, by the definition of $\sqsubseteq$, $e_{11} \sqsubseteq e_{21}$ and $e_{12} \sqsubseteq e_{22}$ then by the induction hypothesis, $e'_{11} \sqsubseteq e'_{21}$ and $PM_1 \sqsubseteq PM_2$ and $e'_{12} \sqsubseteq e'_{22}$ and $T'_{11} \cap \ldots \cap T'_{1n} \sqsubseteq T'_{21} \cap \ldots \cap T'_{2n}$. By the definition of $\rhd$, we have that $PM_1 = T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1$ and $PM_2 = T_{21} \cap \ldots \cap T_{2n} \rightarrow T_2$ and so $T_{11} \cap \ldots \cap T_{1n} \rightarrow T_1 \sqsubseteq T_{21} \cap \ldots \cap T_{2n} \rightarrow T_2$ and therefore by the definition of $\sqsubseteq$, $T_1 \sqsubseteq T_2$. As by the definition of *instances*, S, S, $e \hookrightarrow e$ and $\sqsubseteq$, $e''_{11} \sqsubseteq e''_{21}$ and $e''_{12} \sqsubseteq e''_{22}$, then by the definition of $\sqsubseteq$, $e''_{11}\ e''_{12} \sqsubseteq e''_{21}\ e''_{22}$ and $T_1 \sqsubseteq T_2$.

- Rule C-Gen. If $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_{11} \cap \ldots \cap T_{1n}$ and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_{21} \cap \ldots \cap T_{2n}$ and $e_1 \sqsubseteq e_2$ then by rule C-Gen, $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_{11}$ and ... and $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_{1n}$ and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_{21}$ and ... and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_{2n}$. By the induction hypothesis, $e'_1 \sqsubseteq e'_2$ and $T_{11} \sqsubseteq T_{21}$ and ... and $T_{1n} \sqsubseteq T_{2n}$, and therefore by the definition of $\sqsubseteq$, $T_{11} \cap \ldots \cap T_{1n} \sqsubseteq T_{21} \cap \ldots \cap T_{2n}$.

- Rule C-Inst. If $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_{1i}$ and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_{2i}$ and $e_1 \sqsubseteq e_2$ then by rule C-Inst, $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_{11} \cap \ldots \cap T_{1n}$ and $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_{21} \cap \ldots \cap T_{2n}$. By the induction hypothesis, $e'_1 \sqsubseteq e'_2$ and $T_{11} \cap \ldots \cap T_{1n} \sqsubseteq T_{21} \cap \ldots \cap T_{2n}$, and therefore, by the definition of $\sqsubseteq$, $T_{1i} \sqsubseteq T_{2i}$.

□

Corollary 1 (Monotonicity of cast insertion)**.** *Corollary of Theorem 4. If* $\Gamma \vdash_{\cap CC} e_1 \rightsquigarrow e'_1 : T_1$ *and* $\Gamma \vdash_{\cap CC} e_2 \rightsquigarrow e'_2 : T_2$ *and* $e_1 \sqsubseteq e_2$ *then* $e'_1 \sqsubseteq e'_2$.

Theorem 5 (Conservative Extension)**.** *If* e *is fully static, then* $e \longrightarrow_{\cap S} e' \iff e \longrightarrow_{\cap CC} e'$.

Proof. We proceed by induction on the length of the derivation tree of $\longrightarrow_{\cap S}$ and $\longrightarrow_{\cap CC}$ for the right and left direction of the implication, respectively. Base cases:

- Rule E-AppAbs. If $(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v \longrightarrow_{\cap S} [x \mapsto v]e$ and $(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v \longrightarrow_{\cap CC} [x \mapsto v]e$, then it is proved.

Induction step:

- Rule E-App1.
  - If $e_1\ e_2 \longrightarrow_{\cap S} e'_1\ e_2$ then by rule E-App1, $e_1 \longrightarrow_{\cap S} e'_1$. By the induction hypothesis, $e_1 \longrightarrow_{\cap CC} e'_1$. Therefore, by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap CC} e'_1\ e_2$
  - If $e_1\ e_2 \longrightarrow_{\cap CC} e'_1\ e_2$ then by rule E-App1, $e_1 \longrightarrow_{\cap CC} e'_1$. By the induction hypothesis, $e_1 \longrightarrow_{\cap S} e'_1$. Therefore, by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap S} e'_1\ e_2$
- Rule E-App2.
  - If $v_1\ e_2 \longrightarrow_{\cap S} v_1\ e'_2$ then by rule E-App2, $e_2 \longrightarrow_{\cap S} e'_2$. By the induction hypothesis, $e_2 \longrightarrow_{\cap CC} e'_2$. Therefore, by rule E-App2, $v_1\ e_2 \longrightarrow_{\cap CC} v_1\ e'_2$
  - If $v_1\ e_2 \longrightarrow_{\cap CC} v_1\ e'_2$ then by rule E-App2, $e_2 \longrightarrow_{\cap CC} e'_2$. By the induction hypothesis, $e_2 \longrightarrow_{\cap S} e'_2$. Therefore, by rule E-App2, $v_1\ e_2 \longrightarrow_{\cap S} v_1\ e'_2$

□

Lemma 2 (Type preservation of $\longrightarrow_{\cap IC}$)**.** *If* $c \longrightarrow_{\cap IC} c$ *and*

- $\vdash_{\cap IC} c : T$ *then* $\vdash_{\cap IC} c' : T$.

- $initialType(c) = T$ *then* $initialType(c') = T$.

Proof. We proceed by induction on the length of the derivation tree of $\longrightarrow_{\cap IC}$.

Base cases:

- Rule E-PushBlameIC.
  - If $\vdash_{\cap IC} blame\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2} : T_2$ and by rule E-PushBlameIC, $blame\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ T_2\ l_1\ ^{cl_1}$, then by rule T-BlameIC, $\vdash_{\cap IC} blame\ T_I\ T_2\ l_1\ ^{cl_1} : T_2$, then it is proved.
  - By the definition of *initialType*, $initialType(blame\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2}) = T_I$. By rule E-PushBlameIC, $blame\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ T_2\ l_1\ ^{cl_1}$. Since $initialType(blame\ T_I\ T_2\ l_1\ ^{cl_1}) = T_I$, it is proved.
- Rule E-IdentityIC.
  - If $\vdash_{\cap IC} c : T \Rightarrow^l T\ ^{cl} : T$, then by rule T-SingleIC, $\vdash_{\cap IC} c : T$. By rule E-IdentityIC, $c : T \Rightarrow^l T\ ^{cl} \longrightarrow_{\cap IC} c$.
  - By the definitions of *initialType*, $initialType(c : T \Rightarrow^l T\ ^{cl}) = initialType(c)$. By rule E-IdentityIC, $c : T \Rightarrow^l T\ ^{cl} \longrightarrow_{\cap IC} c$.
- Rule E-SucceedIC.
  - If $\vdash_{\cap IC} c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2} : G$, then by rule T-SingleIC, $\vdash_{\cap IC} c : G$. By rule E-SucceedIC, $c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2} \longrightarrow_{\cap IC} c$.
  - Rule E-SucceedIC. By the definition of *initialType*, $initialType(c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2}) = initialType(c)$. By rule E-SucceedIC, $c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2} \longrightarrow_{\cap IC} c$. Therefore it is proved.
- Rule E-FailIC.
  - If $\vdash_{\cap IC} c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2} : G_2$, and by rule E-FailIC, $c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ G_2\ l_2\ ^{cl_1}$ then by rule T-BlameIC, $\vdash_{\cap IC} blame\ T_I\ G_2\ l_2\ ^{cl_1} : G_2$.
  - By the definition of *initialType*, $initialType(c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2}) = T_I$. By rule E-FailIC, $c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ G_2\ l_2\ ^{cl_1}$, then $initialType(blame\ T_I\ G_2\ l_2\ ^{cl_1}) = T_I$.
- Rule E-GroundIC.
  - If $\vdash_{\cap IC} c : T \Rightarrow^l Dyn\ ^{cl} : Dyn$ then by rule T-SingleIC, $\vdash_{\cap IC} c : T$. By rule E-GroundIC, $c : T \Rightarrow^l Dyn\ ^{cl} \longrightarrow_{\cap IC} c : T \Rightarrow^l G\ ^{cl} : G \Rightarrow^l Dyn\ ^{cl}$, then by rule T-SingleIC, $\vdash_{\cap IC} c : T \Rightarrow^l G\ ^{cl} : G \Rightarrow^l Dyn\ ^{cl} : Dyn$.
  - By the definition of *initialType*, $initialType(c : T \Rightarrow^l Dyn\ ^{cl}) = initialType(c)$. By rule E-GroundIC, $c : T \Rightarrow^l Dyn\ ^{cl} \longrightarrow_{\cap IC} c : T \Rightarrow^l G\ ^{cl} : G \Rightarrow^l Dyn\ ^{cl}$, then $initialType(c : T \Rightarrow^l G\ ^{cl} : G \Rightarrow^l Dyn\ ^{cl}) = initialType(c)$.
- Rule E-ExpandIC.

- If $\vdash_{\cap IC} c : Dyn \Rightarrow^{l} T\ ^{cl} : T$ then by rule T-SingleIC, $\vdash_{\cap IC} c : Dyn$. By rule E-ExpandIC, $c : Dyn \Rightarrow^{l} T\ ^{cl} \longrightarrow_{\cap IC} c : Dyn \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} T\ ^{cl}$, then by rule T-SingleIC, $\vdash_{\cap IC} c : Dyn \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} T\ ^{cl} : T$.
- By the definition of *initialType*, $initialType(c : Dyn \Rightarrow^{l} T\ ^{cl}) = initialType(c)$. By rule E-ExpandIC, $c : Dyn \Rightarrow^{l} T\ ^{cl} \longrightarrow_{\cap IC} c : Dyn \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} T\ ^{cl}$. Since $initialType(c : Dyn \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} T\ ^{cl}) = initialType(c)$, it is proved.

Induction step:

- Rule E-EvaluateIC.
  - If $\vdash_{\cap IC} c : T_1 \Rightarrow^{l} T_2\ ^{cl} : T_2$ then by rule T-SingleIC, $\vdash_{\cap IC} c : T_1$. By rule E-EvaluateIC, $c \longrightarrow_{\cap IC} c'$. By the induction hypothesis, $\vdash_{\cap IC} c' : T_1$. By rule E-EvaluateIC, $c : T_1 \Rightarrow^{l} T_2\ ^{cl} \longrightarrow_{\cap IC} c' : T_1 \Rightarrow^{l} T_2\ ^{cl}$, then by rule T-SingleIC, $\vdash_{\cap IC} c' : T_1 \Rightarrow^{l} T_2\ ^{cl} : T_2$.
  - By the definition of *initialType*, $initialType(c : T_1 \Rightarrow^{l} T_2\ ^{cl}) = initialType(c)$. By rule E-EvaluateIC, $c \longrightarrow_{\cap IC} c'$. By the induction hypothesis, $initialType(c') = initialType(c)$. By rule E-EvaluateIC, $c : T_1 \Rightarrow^{l} T_2\ ^{cl} \longrightarrow_{\cap IC} c' : T_1 \Rightarrow^{l} T_2\ ^{cl}$. Since $initialType(c' : T_1 \Rightarrow^{l} T_2\ ^{cl}) = initialType(c')$, it is proved.

□

Lemma 3 (Progress of $\longrightarrow_{\cap IC}$). *If $\Gamma \vdash_{\cap IC} c : T$ and $initialType(c) = T_I$ then either c is a cast value or there exists a c' such that $c \longrightarrow_{\cap IC} c'$.*

Proof. We proceed by induction on the length of the derivation tree of $\vdash_{\cap IC} c : T$.

Base cases:

- Rule T-BlameIC. As $\vdash_{\cap IC} blame\ T_I\ T_F\ l\ ^{cl} : T_F$, $initialType(blame\ T_I\ T_F\ l\ ^{cl}) = T_I$ and $blame\ T_I\ T_F\ l\ ^{cl}$ is a cast value, it is proved.
- Rule T-EmptyIC. As $\vdash_{\cap IC} \varnothing\ T\ ^{cl} : T$, $initialType(\varnothing\ T\ ^{cl}) = T$ and $\varnothing\ T\ ^{cl}$ is a cast value, it is proved.

Induction step:

- Rule T-SingleIC. If $\vdash_{\cap IC} c : T_1 \Rightarrow^{l} T_2\ ^{c}l : T_2$ and $initialType(c : T_1 \Rightarrow^{l} T_2\ ^{c}l) = T_I$ then by rule T-SingleIC, $\vdash_{\cap IC} c : T_1$ and $initialType(c) = T_I$. By the induction hypothesis, either c is a cast value or there is a c' such that $c \longrightarrow_{\cap IC} c'$. If c is a cast value, then c can either be of the form $blame\ T_I\ T_F\ l\ ^{cl}$, in which case by rule E-PushBlameIC, $blame\ T_I\ T_F\ l_1\ ^{cl_1} : T_1 \Rightarrow^{l_2} T_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ T_2\ l_1\ ^{cl_1}$ or c is a cast value 1 or is an empty cast. If c is a cast value 1 or is an empty cast then $c : T_1 \Rightarrow^{l} T_2\ ^{c}l$ can be of one of the folowing forms:
  - $c : T \Rightarrow^{l} T\ ^{cl}$. Then by rule E-IdentityIC, $c : T \Rightarrow^{l} T\ ^{cl} \longrightarrow_{\cap IC} c$.
  - $c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2}$. Then by rule E-SucceedIC, $c : G \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G\ ^{cl_2} \longrightarrow_{\cap IC} c$.
  - $c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2}$. Then by rule E-FailIC, $c : G_1 \Rightarrow^{l_1} Dyn\ ^{cl_1} : Dyn \Rightarrow^{l_2} G_2\ ^{cl_2} \longrightarrow_{\cap IC} blame\ T_I\ G_2\ l_2\ ^{cl_1}$.
  - $c : T \Rightarrow^{l} Dyn\ ^{cl}$. Then by rule E-GroundIC, $c : T \Rightarrow^{l} Dyn\ ^{cl} \longrightarrow_{\cap IC} c : T \Rightarrow^{l} G\ ^{cl} : G \Rightarrow^{l} Dyn\ ^{cl}$.

– $c : Dyn \Rightarrow^l T\ ^{cl}$. Then by rule E-ExpandIC, $c : Dyn \Rightarrow^l T\ ^{cl} \longrightarrow_{\cap IC} c : Dyn \Rightarrow^l G\ ^{cl} : G \Rightarrow^l T\ ^{cl}$.

If there is a c' such that $c \longrightarrow_{\cap IC} c'$, then by rule E-EvaluateIC, $c : T_1 \Rightarrow^l T_2\ ^c l \longrightarrow_{\cap IC} c' : T_1 \Rightarrow^l T_2\ ^c l$.

$\square$

Lemma 4 (Type preservation of $\longrightarrow_{\cap CC}$). *Depends on Lemmas 2 and 3. If $\Gamma \vdash_{\cap CC} e : T_1 \cap \ldots \cap T_n$ and $e \longrightarrow_{\cap CC} e'$ then $\Gamma \vdash_{\cap CC} e' : T_1 \cap \ldots \cap T_m$ such that $m \leq n$.*

Proof. We proceed by induction on the length of the derivation tree of $\longrightarrow_{\cap CC}$.

Base cases:

- Rule E-PushBlame1. If $\Gamma \vdash_{\cap CC} blame_{T_2}\ l\ e_2 : T_1$ and $blame_{T_2}\ l\ e_2 \longrightarrow_{\cap CC} blame_{T_1}\ l$ then by rule T-Blame, $\Gamma \vdash_{\cap CC} blame_{T_1}\ l : T_1$.

- Rule E-PushBlame2. If $\Gamma \vdash_{\cap CC} e_1\ blame_{T_2}\ l : T_1$ and $e_1\ blame_{T_2}\ l \longrightarrow_{\cap CC} blame_{T_1}\ l$ then by rule T-Blame, $\Gamma \vdash_{\cap CC} blame_{T_1}\ l : T_1$.

- Rule E-PushBlameCast. If $\Gamma \vdash_{\cap CC} blame_T\ l : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n$ and $blame_T\ l : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} blame_{T_1 \cap \ldots \cap T_n}\ l$ then by rule T-Blame, $\Gamma \vdash_{\cap CC} blame_{T_1 \cap \ldots \cap T_n}\ l : T_1 \cap \ldots \cap T_n$.

- Rule E-AppAbs. There exists a type $T_1 \cap \ldots \cap T_n$ such that we can deduce $\Gamma \vdash_{\cap CC} (\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v : T$ from $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_1 \cap \ldots \cap T_n \rightarrow T$ and $\Gamma \vdash_{\cap CC} v : T_1 \cap \ldots \cap T_n$ (x does not occur in Γ). Moreover, $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_1 \cap \ldots \cap T_n \rightarrow T$ only if $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap CC} e : T$. By rule E-AppAbs, $(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v \longrightarrow_{\cap CC} [x \mapsto v]e$. To obtain $\Gamma \vdash_{\cap CC} [x \mapsto v]e : T$, it is sufficient to replace, in the proof of $\Gamma, x : T_1 \cap \ldots \cap T_n \vdash_{\cap CC} e : T$, the statements $x : T_i$ (introduzed by the rules T-Var and T-Inst) by the deductions of $\Gamma \vdash_{\cap CC} v : T_i$ for $1 \leq i \leq n$. (Proof adapted from [1])

- Rule E-SimulateArrow. If $\Gamma \vdash_{\cap CC} (v_1 : cv_1 \cap \ldots \cap cv_n)\ v_2 : T_{12} \cap \ldots \cap T_{n2}$, then by rule T-App', $\Gamma \vdash_{\cap CC} v_1 : cv_1 \cap \ldots \cap cv_n : T_1 \cap \ldots \cap T_n$ such that $\exists i \in 1..n\ .\ T_i = T_{i1} \rightarrow T_{i2}$ and $\Gamma \vdash_{\cap CC} v_2 : T_{11} \cap \ldots \cap T_{n1}$. As $\Gamma \vdash_{\cap CC} v_1 : cv_1 \cap \ldots \cap cv_n : T_1 \cap \ldots \cap T_n$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v_1 : T''_1 \cap \ldots \cap T''_l$ and $\vdash_{\cap IC} cv_1 : T_1$ and ... and $\vdash_{\cap IC} cv_n : T_n$ and $I_1 = initialType(cv_1)$ and ... and $I_n = initialType(cv_n)$ such that $\{I_1, \ldots, I_n\} \subseteq \{T''_1, \ldots, T''_l\}$ and $I_1 \cap \ldots \cap I_n = T''_1 \cap \ldots \cap T''_n$ and $n \leq l$. For the sake of simplicity lets elide cast labels and blame labels. By the definition of SimulateArrow, we have that $c'_1 = c''_1 : T'_{11} \rightarrow T'_{12} \Rightarrow T_{11} \rightarrow T_{12}$ and ... and $c'_m = c''_m : T'_{m1} \rightarrow T'_{m2} \Rightarrow T_{m1} \rightarrow T_{m2}$, for some $m \leq n$. Also, $c_{11} = \varnothing\ T_{11}\ : T_{11} \Rightarrow T'_{11}$ and ... and $c_{m1} = \varnothing\ T_{m1}\ : T_{m1} \Rightarrow T'_{m1}$ and $c_{12} : \varnothing\ T'_{12}\ : T'_{12} \Rightarrow T_{12}$ and ... and $c_{m2} = \varnothing\ T'_{m2}\ : T'_{m2} \Rightarrow T_{m2}$ and $initialType(c^s_1) = I_1$ and ... and $initialType(c^s_m) = I_m$ and $\vdash_{\cap IC} c^s_1 : T'_{11} \rightarrow T'_{12}$ and ... and $\vdash_{\cap IC} c^s_m : T'_{m1} \rightarrow T'_{m2}$. As by rule T-Gen and T-Inst $\Gamma \vdash_{\cap CC} v_1 : T''_1 \cap \ldots \cap T''_m$ and $I_1 \cap \ldots \cap I_m = T''_1 \cap \ldots \cap T''_m$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v_1 : c^s_1 \cap \ldots \cap c^s_m : T'_{11} \rightarrow T'_{12} \cap \ldots \cap T'_{m1} \rightarrow T'_{m2}$. As by rule T-Gen and T-Inst $\Gamma \vdash_{\cap CC} v_2 : T_{11} \cap \ldots \cap T_{m1}$ and $\vdash_{\cap IC} c_{11} : T'_{11}$ and ... and $\vdash_{\cap IC} c_{m1} : T'_{m1}$ and $initialType(c_{11}) = T_{11}$ and ... and $initialType(c_{m1}) = T_{m1}$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v_2 : c_{11} \cap \ldots \cap c_{m1} : T'_{11} \cap \ldots \cap T'_{m1}$. Therefore, by rule T-App', $\Gamma \vdash_{\cap CC} (v_1 : c^s_1 \cap \ldots \cap c^s_m)\ (v_2 : c_{11} \cap \ldots \cap c_{m1}) : T'_{12} \cap \ldots \cap T'_{m2}$. As $\vdash_{\cap IC} c_{12} : T_{12}$ and ... and $\vdash_{\cap IC} c_{m2} : T_{m2}$ and $initialType(c_{12}) = T'_{12}$ and ... and $initialType(c_{m2}) = T'_{m2}$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} (v_1 : c^s_1 \cap \ldots \cap c^s_m)\ (v_2 : c_{11} \cap \ldots \cap c_{m1}) : c_{12} \cap \ldots \cap c_{m2} : T_{12} \cap \ldots \cap T_{m2}$.

By rule E-SimulateArrow, $(v_1 : cv_1 \cap \ldots \cap cv_n)\ v_2 \longrightarrow_{\cap CC}$ $(v_1 : c^s_1 \cap \ldots \cap c^s_m)\ (v_2 : c_{11} \cap \ldots \cap c_{m1}) : c_{12} \cap \ldots \cap c_{m2}$, therefore it is proved.

- Rule E-MergeCasts. If $\Gamma \vdash_{\cap CC} v : cv_1 \cap \ldots \cap cv_n : c'_1 \cap \ldots \cap c'_m : F'_1 \cap \ldots \cap F'_m$ then by rule T-IntersectionCasts, $\Gamma \vdash_{\cap CC} v : cv_1 \cap \ldots \cap cv_n : F_1 \cap \ldots \cap F_n$ and $\vdash_{\cap IC} c'_1 : F'_1$ and ... and $\vdash_{\cap IC} c'_m : F'_m$ and $initialType(c'_1) = I'_1$ and $initialType(c'_m) = I'_m$ such that $\{I'_1, \ldots, I'_m\} \subseteq \{F_1, \ldots, F_n\}$ and $I'_1 \cap \ldots \cap I'_m = F_1 \cap \ldots \cap F_m$ and $m \leq n$. As $\Gamma \vdash_{\cap CC} v : cv_1 \cap \ldots \cap cv_n : F_1 \cap \ldots \cap F_n$ then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v : T_1 \cap \ldots \cap T_l$ and $\vdash_{\cap IC} cv_1 : F_1$ and ... and $\vdash_{\cap IC} cv_n : F_n$ and $initialType(cv_1) : I_1$ and ... and $initialType(cv_n) : I_n$ such that $\{I_1, \ldots, I_n\} \subseteq \{T_1, \ldots, T_l\}$ and $I_1 \cap \ldots \cap I_n \subseteq T_1 \cap \ldots \cap T_n$ and $n \leq l$. By the definition of mergeCasts, $\vdash_{\cap IC} c''_1 : F''_1$ and ... and $\vdash_{\cap IC} c''_j : F''_j$ and $initialType(c''_1) = I''_1$ and ... and $initialType(c''_j) = I''_j$ such that $\{I''_1, \ldots, I''_j\} \subseteq \{T_1, \ldots, T_l\}$ and $I''_1 \cap \ldots \cap I''_j = T_1 \cap \ldots \cap T_j$ and $\{F''_1, \ldots, F''_j\} \subseteq \{F'_1, \ldots, F'_m\}$ and $F''_1 \cap \ldots \cap F''_j = F'_1 \cap \ldots \cap F'_j$ and $j \leq l$ and $j \leq m$. By rule T-Gen and T-Inst, $\Gamma \vdash_{\cap CC} v : T_1 \cap \ldots \cap T_j$ and therefore by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v : c''_1 \cap \ldots \cap c''_j : F''_1 \cap \ldots \cap F''_j$. By rule E-MergeCasts, $v : cv_1 \cap \ldots \cap cv_n : c'_1 \cap \ldots \cap c'_m \longrightarrow_{\cap CC} v : c''_1 \cap \ldots \cap c''_j$.
- Rule E-EvaluateCasts. If $\Gamma \vdash_{\cap CC} v : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n$ then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v : T'_1 \cap \ldots \cap T'_n$ and $\vdash_{\cap IC} c_1 : T_1$ and ... and $\vdash_{\cap IC} c_n : T_n$ and $I_1 = initialType(c_1)$ and ... and $I_n = initialType(c_n)$ and $I_1 \cap \ldots \cap I_n = T'_1 \cap \ldots \cap T'_n$. By rule E-EvaluateCasts, $c_1 \longrightarrow_{\cap IC} cv_1$ and ... and $c_n \longrightarrow_{\cap IC} cv_n$. By Lemmas 2 and 3, $\vdash_{\cap IC} cv_1 : T_1$ and $initialType(cv_1) = I_1$ and ... and $\vdash_{\cap IC} cv_n : T_n$ and $initialType(cv_n) = I_n$. Therefore by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v : cv_1 \cap \ldots \cap cv_n : T_1 \cap \ldots \cap T_n$. By rule E-EvaluateCasts, $v : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} v : cv_1 \cap \ldots \cap cv_n$.
- Rule E-PropagateBlame. If $\Gamma \vdash_{\cap CC} v : blame\ T'_1\ T_1\ l_1\ {}^{m_1} \cap \ldots \cap blame\ T'_n\ T_n\ l_n\ {}^{m_n} : T_1 \cap \ldots \cap T_n$ and by rule E-PropagateBlame $v : blame\ T'_1\ T_1\ l_1\ {}^{m_1} \cap \ldots \cap blame\ T'_n\ T_n\ l_n\ {}^{m_n} \longrightarrow_{\cap CC} blame_{(T_1 \cap \ldots \cap T_n)}\ l_1$, then by rule T-Blame, $\Gamma \vdash_{\cap CC} blame_{(T_1 \cap \ldots \cap T_n)}\ l_1 : T_1 \cap \ldots \cap T_n$.
- Rule E-RemoveEmpty. If $\Gamma \vdash_{\cap CC} v : \varnothing\ T_1\ {}^{m_1} \cap \ldots \cap \varnothing\ T_n\ {}^{m_n} : T_1 \cap \ldots \cap T_n$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} v : T_1 \cap \ldots \cap T_n$ and $\vdash_{\cap IC} \varnothing\ T_1\ {}^{m_1} : T_1$ and ... and $\vdash_{\cap IC} \varnothing\ T_n\ {}^{m_n} : T_n$ and $initialType(\varnothing\ T_1\ {}^{m_1}) = T_1$ and ... and $initialType(\varnothing\ T_n\ {}^{m_n}) = T_n$. Therefore, by rule E-RemoveEmpty, $v : \varnothing\ T_1\ {}^{m_1} \cap \ldots \cap \varnothing\ T_n\ {}^{m_n} \longrightarrow_{\cap CC} v$.

Induction step:

- Rule E-App1. There are two possibilities:
  - If $\Gamma \vdash_{\cap CC} e_1\ e_2 : T$, then by rule T-App, $\Gamma \vdash_{\cap CC} e_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap CC} e_2 : T_1 \cap \ldots \cap T_n$. By rule E-App1, $e_1 \longrightarrow_{\cap IC} e'_1$, so by the induction hypothesis, $\Gamma \vdash_{\cap CC} e'_1 : T_1 \cap \ldots \cap T_n \to T$. As by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap IC} e'_1\ e_2$, then by rule T-App, $\Gamma \vdash_{\cap CC} e'_1\ e_2 : T$.
  - If $\Gamma \vdash_{\cap CC} e_1\ e_2 : T_{12} \cap \ldots \cap T_{n2}$, then by rule T-App', $\Gamma \vdash_{\cap CC} e_1 : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}$ and $\Gamma \vdash_{\cap CC} e_2 : T_{11} \cap \ldots \cap T_{n1}$. By rule E-App1, $e_1 \longrightarrow_{\cap IC} e'_1$, so by the induction hypothesis, $\Gamma \vdash_{\cap CC} e'_1 : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}$. As by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap IC} e'_1\ e_2$, then by rule T-App', $\Gamma \vdash_{\cap CC} e'_1\ e_2 : T_{12} \cap \cdots \cap T_{n2}$.
- Rule E-App2. There are two possibilities:
  - If $\Gamma \vdash_{\cap CC} v_1\ e_2 : T$, then by rule T-App, $\Gamma \vdash_{\cap CC} v_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap CC} e_2 : T_1 \cap \ldots \cap T_n$. By rule E-App2, $e_2 \longrightarrow_{\cap IC} e'_2$, so by the induction hypothesis, $\Gamma \vdash_{\cap CC} e'_2 : T_1 \cap \ldots \cap T_n$. As by rule E-App2, $v_1\ e_2 \longrightarrow_{\cap IC} v_1\ e'_2$, then by rule T-App, $\Gamma \vdash_{\cap CC} v_1\ e'_2 : T$.

- If $\Gamma \vdash_{\cap CC} v_1\ e_2 : T_{12} \cap \ldots \cap T_{n2}$, then by rule T-App', $\Gamma \vdash_{\cap CC} v_1 : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}$ and $\Gamma \vdash_{\cap CC} e_2 : T_{11} \cap \ldots \cap T_{n1}$. By rule E-App2, $e_2 \longrightarrow_{\cap IC} e_2'$, so by the induction hypothesis, $\Gamma \vdash_{\cap CC} e_2' : T_{11} \cap \ldots \cap T_{n1}$. As by rule E-App1, $v_1\ e_2 \longrightarrow_{\cap IC} v_1\ e_2'$, then by rule T-App', $\Gamma \vdash_{\cap CC} v_1\ e_2' : T_{12} \cap \cdots \cap T_{n2}$..

- Rule E-Evaluate. If $\Gamma \vdash_{\cap CC} e : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} e : T, \vdash_{\cap IC} c_1 : T_1$ and ... and $\vdash_{\cap IC} c_n : T_n$ and $initialType(c_1) \cap \ldots \cap initialType(c_n) = T$. By rule E-Evaluate, $e \longrightarrow_{\cap IC} e'$, so by the induction hypothesis, $\Gamma \vdash_{\cap CC} e' : T$. As by rule E-Evaluate, $e : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap IC} e' : c_1 \cap \ldots \cap c_n$, then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} e' : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n$.

□

Lemma 5 (Progress of $\longrightarrow_{\cap CC}$)**.** *If $\Gamma \vdash_{\cap CC} e : T$ then either e is a value or there exists an e' such that $e \longrightarrow_{\cap CC} e'$.*

Proof. We proceed by induction on the length of the derivation tree of $\Gamma \vdash_{\cap CC} e : T$.

Base cases:

- Rule T-Var. If $\Gamma \vdash_{\cap CC} x : T$, then x is a value.
- Rule T-Int. If $\Gamma \vdash_{\cap CC} n : Int$ then n is a value.
- Rule T-True. If $\Gamma \vdash_{\cap CC} true : Bool$ then $true$ is a value.
- Rule T-False. If $\Gamma \vdash_{\cap CC} false : Bool$ then $false$ is a value.

Induction step:

- Rule T-Abs. If $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_1 \cap \ldots \cap T_n \to T$ then $\lambda x : T_1 \cap \ldots \cap T_n\ .\ e$ is a value.
- Rule T-Abs'. If $\Gamma \vdash_{\cap CC} \lambda x : T_1 \cap \ldots \cap T_n\ .\ e : T_i \to T$ then $\lambda x : T_1 \cap \ldots \cap T_n\ .\ e$ is a value.
- Rule T-App. If $\Gamma \vdash_{\cap CC} e_1\ e_2 : T$ then by rule T-App, $\Gamma \vdash_{\cap CC} e_1 : T_1 \cap \ldots \cap T_n \to T$ and $\Gamma \vdash_{\cap CC} e_2 : T_1 \cap \ldots \cap T_n$. By the induction hypothesis, e_1 is either a value or there is a e_1' such that $e_1 \longrightarrow_{\cap CC} e_1'$ and e_2 is either a value or there is a e_2' such that $e_2 \longrightarrow_{\cap CC} e_2'$. If e_1 is a value, then by rule E-PushBlame1, $(blame_{T_2}\ l)\ e_2 \longrightarrow_{\cap CC} blame_{T_1}\ l$. If e_2 is a value, then by rule E-PushBlame2, $e_1\ (blame_{T_2}\ l) \longrightarrow_{\cap CC} blame_{T_1}\ l$. If e_1 is not a value, then by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap CC} e_1'\ e_2$. If e_1 is a value and e_2 is not a value, then by rule E-App2, $v_1\ e_2 \longrightarrow_{\cap CC} v_1\ e_2'$. If both e_1 and e_2 are values then e_1 must be an abstraction $(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)$, and by rule E-AppAbs $(\lambda x : T_1 \cap \ldots \cap T_n\ .\ e)\ v_2 \longrightarrow_{\cap CC} [x \mapsto v_2]e$.
- Rule T-Gen. If $\Gamma \vdash_{\cap CC} e : T_1 \cap \ldots \cap T_n$ then by rule T-Gen, $\Gamma \vdash_{\cap CC} e : T_1$ and ... and $\Gamma \vdash_{\cap CC} e : T_n$. By the induction hypothesis, either e is a value or there exists an e' such that $e \longrightarrow_{\cap CC} e'$.
- Rule T-Inst. If $\Gamma \vdash_{\cap CC} e : T_i$ then by rule T-Inst, $\Gamma \vdash_{\cap CC} e : T_1 \cap \ldots \cap T_n$, such that $T_i \in \{T_1, \ldots, T_n\}$. By the induction hypothesis, either e is a value or there exists an e' such that $e \longrightarrow_{\cap CC} e'$.

- Rule T-App'. If $\Gamma \vdash_{\cap CC} e_1\ e_2 : T_{12} \cap \ldots \cap T_{n2}$ then by rule T-App', $\Gamma \vdash_{\cap CC} e_1 : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}$ and $\Gamma \vdash_{\cap CC} e_2 : T_{11} \cap \ldots \cap T_{n1}$. By the induction hypothesis, e_1 is either a value or there is a e_1' such that $e_1 \longrightarrow_{\cap CC} e_1'$ and e_2 is either a value or there is a e_2' such that $e_2 \longrightarrow_{\cap CC} e_2'$. If e_1 is a value, then by rule E-PushBlame1, $(blame_{T_2}\ l)\ e_2 \longrightarrow_{\cap CC} blame_{T_1}\ l$. If e_2 is a value, then by rule E-PushBlame2, $e_1\ (blame_{T_2}\ l) \longrightarrow_{\cap CC} blame_{T_1}\ l$. If e_1 is not a value, then by rule E-App1, $e_1\ e_2 \longrightarrow_{\cap CC} e_1'\ e_2$. If e_1 is a value and e_2 is not a value, then by rule E-App2, $v_1\ e_2 \longrightarrow_{\cap CC} v_1\ e_2'$. If both e_1 and e_2 are values then e_1 must be an abstraction $(\lambda x : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}.\ e)$, and by rule E-AppAbs $(\lambda x : T_{11} \to T_{12} \cap \ldots \cap T_{n1} \to T_{n2}\ .\ e)\ v_2 \longrightarrow_{\cap CC} [x \mapsto v_2]e$.
- Rule T-IntersectionCast. If $\Gamma \vdash_{\cap CC} e : c_1 \cap \ldots \cap c_n : T_1 \cap \ldots \cap T_n$ then by rule T-IntersectionCast, $\Gamma \vdash_{\cap CC} e : T$. By the induction hypothesis, e is either a value, or there is an e' such that $e \longrightarrow_{\cap CC} e'$. If e is a value, then either by rule E-EvaluateCasts, $v : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} v : cv_1 \cap \ldots \cap cv_n$, or by rule E-PushBlameCast, $blame_T\ l : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} blame_{T_1 \cap \ldots \cap T_n}\ l$. If there is an e' such that $e \longrightarrow_{\cap CC} e'$, then by rule E-Evaluate, $e : c_1 \cap \ldots \cap c_n \longrightarrow_{\cap CC} e' : c_1 \cap \ldots \cap c_n$.
- Rule T-Blame. If $\Gamma \vdash_{\cap CC} blame_T\ l : T$ then $blame_T\ l$ is a value.

$\square$

Theorem 6 (Type Safety of $\longrightarrow_{\cap CC}$)**.** *Depends on Lemmas 4 and 5. Both Type Preservation and Progress hold for $\longrightarrow_{\cap CC}$.*

Proof. We have Type Preservation (by Lemma 4) and Progress (by Lemma 5) for $\longrightarrow_{\cap CC}$. $\square$

Theorem 7 (Blame Theorem)**.** *If $\Gamma \vdash_{\cap CC} e : T$ and $e \longrightarrow^*_{\cap CC} blame_T\ l$ then l is not a safe cast of e.*

Theorem 8 (Gradual Guarantee)**.** *If $\Gamma \vdash_{\cap CC} e_1 : T_1$ and $\Gamma \vdash_{\cap CC} e_2 : T_2$ and $e_1 \sqsubseteq e_2$ then:*

1. *if $e_2 \longrightarrow_{\cap CC} e_2'$ then $e_1 \longrightarrow^*_{\cap CC} e_1'$ and $e_1' \sqsubseteq e_2'$.*
2. *if $e_1 \longrightarrow_{\cap CC} e_1'$ then either $e_2 \longrightarrow^*_{\cap CC} e_2'$ and $e_1' \sqsubseteq e_2'$ or $e_2 \longrightarrow^*_{\cap CC} blame_{T_2}\ l$.*

References

[1] Mario Coppo, Mariangiola Dezani-Ciancaglini, et al. An extension of the basic functionality theory for the λ-calculus. *Notre Dame journal of formal logic*, 21(4):685–693, 1980.