

Manual de Hue

Nicolas Cabral

January 27, 2017

Abstract

The abstract text goes here.

1 Hue

Es un asistente de pruebas basados en el Calculo de Construccions λC

2 PTS: Pure Type System

Definición 2.1 (PTS). Un PTS \mathcal{P} esta definido por $\{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$ donde

- \mathcal{S} es un conjunto sorts
- \mathcal{V} es un conjunto de variables
- \mathcal{A} es un conjunto no vacio de $\mathcal{S} \times \mathcal{S}$ llamados axiomas
- \mathcal{R} es un conjunto de $\mathcal{S} \times \mathcal{S} \times \mathcal{S}$ llamadas Π – *Reglas*

Definición 2.2 (Pseudotérminos). El conjunto \mathcal{T} de pseudotérminos de un PTS $\mathcal{P} = \{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$ es el menor conjunto que satisface lo siguiente:

- $\mathcal{S} \cup \mathcal{V} \subset \mathcal{T}$
- Si $a \in \mathcal{T}$ y $b \in \mathcal{T}$ entonces $ab \in \mathcal{T}$
- Si $A \in \mathcal{T}$, $B \in \mathcal{T}$ y $x \in \mathcal{V}$ entonces $(\lambda x : A.B) \in \mathcal{T}$
- Si $A \in \mathcal{T}$, $B \in \mathcal{T}$ y $x \in \mathcal{V}$ entonces $(\Pi x : A.B) \in \mathcal{T}$

Definición 2.3 (PTS funcional). Un PTS se dice *funcional* si

- $\langle s_1, s \rangle \in \mathcal{A}$ y $\langle s_1, s' \rangle \in \mathcal{A}$ implica $s = s'$;
- $\langle s_1, s_2, s \rangle \in \mathcal{R}$ y $\langle s_1, s_2, s' \rangle \in \mathcal{R}$ implica $s = s'$.

Definición 2.4 (PTS full). Un PTS se dice *full* si para todo $s_1, s_2 \in \mathcal{S}$ existe un $s_3 \in \mathcal{S}$ con $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$.

Definición 2.5 (Relación \vdash). Definimos a la relacion \vdash , con $\vdash \subseteq \mathcal{C} \times \mathcal{T} \times \mathcal{T}$ como la menor relación que cumple:

(Srt)	$\overline{\emptyset \vdash s_1 : s_2}$	$(s_1, s_2) \in \mathcal{A}$
(Var)	$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}$	
(Wk)	$\frac{\Gamma \vdash b : B \quad \Gamma \vdash A : s}{\Gamma, x : A \vdash b : B}$	$b \in \mathcal{S} \cup \mathcal{V}$
(Pi)	$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash \Pi x : A. B : s_3}$	$(s_1, s_2, s_3) \in \mathcal{R}$
(Lda)	$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash b : B \quad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash \lambda a : A. b : \Pi x : A. B}$	$(s_1, s_2, s_3) \in \mathcal{R}$
(App)	$\frac{\Gamma \vdash a : \Pi x : B. A \quad \Gamma \vdash b : B}{\Gamma \vdash ab : A[x := b]}$	
(Cnv)	$\frac{\Gamma \vdash a : A \quad \Gamma \vdash B : s}{\Gamma \vdash a : B}$	$A \simeq B$

3 Teoría del calculo de λC

Definición 3.1. λC es un PTS definido por $\{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$ tales que:

- $\mathcal{S} = \{Prop, Type\}$
- $\mathcal{V} = \dots$
- $\mathcal{A} = \{(Prop, Type)\}$
- $\mathcal{R} = \{(Prop, Prop, Prop), (Type, Prop, Prop), (Type, Type, Type)\}$

Lema 3.1. λC es un PTS *full*

Proof.

□

3.1 Terminos

3.2 Reduccion

3.3 Contextos

3.4 Typechecker

[3] [1] [2]

$$\frac{}{\vdash s_1 : s_2} \text{Srtnsd} \quad (1)$$

$$\frac{\Gamma \vdash A : \Rightarrow s}{\Gamma, x : A \vdash_{nsd} x : A} \text{Varnsd} \quad (2)$$

$$\frac{\Gamma \vdash_{nsd} b : B \quad \Gamma \vdash A : \Rightarrow s}{\Gamma, x : A \vdash_{nsd} b : B} \text{Wknsd} \quad (3)$$

$$\frac{\Gamma \vdash A : \Rightarrow s_1 \quad \Gamma, x : A \vdash_{nsd} B : \Rightarrow s_2}{\Gamma \vdash_{nsd} \Pi x : A. B : s_3} \text{Pinsd} \quad (4)$$

$$\frac{\Gamma \vdash A : \Rightarrow s_1 \quad \Gamma, x : A \vdash_{nsd} b : \Rightarrow B \quad \Gamma, x : A \vdash_{nsd} B : \Rightarrow s_2}{\Gamma \vdash_{nsd} \lambda x : A. b : \Pi x : A. B} \text{Ldansd} \quad (5)$$

$$\frac{\Gamma \vdash a : \Rightarrow \Pi x : A. B \quad \Gamma \vdash_{nsd} b : \Rightarrow B}{\Gamma \vdash_{nsd} a b : A[x := b]} \text{Appnsd} \quad (6)$$

4 Conclusion

Write your conclusion here.

References

- [1] L. S. van Benthem Jutting, J. McKinna, and R. Pollack. Checking algorithms for pure type systems. In H. Barendregt and T. Nipkow, editors, *Types for Proofs and Programs*, pages 19–61. Springer, Berlin, Heidelberg, 1993.
- [2] L. S. van Benthem Jutting, James McKinna, and Robert Pollack. Checking algorithms for pure type systems. In Henk Barendregt and Tobias Nipkow, editors, *Types for Proofs and Programs, International Workshop TYPES'93, Nijmegen, The Netherlands, May 24-28, 1993, Selected Papers*, volume 806 of *Lecture Notes in Computer Science*, pages 19–61. Springer, 1993.
- [3] L.S. van Benthem Jutting, J. McKinna, and R. Pollack. Checking algorithms for pure type systems, 1993.