The Cedilleum Language Specification

Syntax, Typing, Reduction, and Elaboration

Christopher Jenkins

June 18, 2018

1 Syntax

id			identifiers for definitions
u			term variables
X			type variables
k			kind variables
x	$::=$	$id \mid u \mid X$	non-kind variables

Figure 1: Identifiers

$$
\begin{array}{rcl}
uterms & ::= & u \\
 & & \lambda\, u.\, uterm \\
 & & uterm\ uterm
\end{array}
$$

Figure 2: Untyped terms

mod	::=	**module** id **.** $imprt^*$ cmd^*	module declarations
$imprt$	::=	**import** id **.**	module imports
cmd	::=	$defTermOrType$	definitions
		$defDataType$	
		$defKind$	
$defTermOrType$	::=	id $checkType^?$ $=$ $term$ **.**	term definition
		id **:** $kind$ $=$ $type$ **.**	type definition
$defDataType$	::=	**data** id $param^*$ **:** $kind$ $=$ $constr^*$ **.**	datatype definitions
$defKind$	::=	k $=$ $kind$	kind definition
$checkType$	::=	**:** $type$	annotation for term definition
$param$	::=	$(x$ **:** $typeOrKind)$	
$typeOrKind$	::=	$type$	
		$kind$	
$constr$	::=	$\mid$ id **:** $type$	

Figure 3: Modules and definitions

$kind$			
	::=	$\Pi\ x$ **:** $typeOrKind$ **.** $kind$	explicit product
		$typeOrKind \to kind$	kind arrow
		$\star$	
		$(kind)$	
$type$			
	::=	$\Pi\ x$ **:** $type$ **.** $type$	explicit product
		$\forall\ x$ **:** $typeOrKind$ **.** $type$	implicit product
		$\lambda\ x$ **:** $typeOrKind$ **.** $type$	type-level function
		$type \Rightarrow type$	arrow with erased domain
		$type \to type$	normal arrow type
		$type \cdot type$	application to another type
		$type\ term$	application to a term
		$\{\ uterm\ \simeq\ uterm\}$	untyped equality
		$(type)$	
		X	type variable
		$\bullet$	hole for incomplete types

Figure 4: Kinds and types

$term$	::=	$\lambda\ x\ class^?\ .\ term$	normal abstraction
		$\Lambda\ x\ class^?\ .\ term$	erased abstraction
		$[\ defTermOrType\ ]\ \text{-}\ term$	let
		$\rho\ term\ \text{-}\ term$	equality elimination by rewriting
		$\phi\ term\ \text{-}\ term\ \{term\}$	type cast
		$\chi\ type^?\ \text{-}\ term$	check a term against a type
		$\delta\ \text{-}\ term$	ex falso quodlibet
		$\theta\ term\ term$	elimination with a motive
		$term\ term$	applications
		$term\ \text{-}\ term$	application to an erased term
		$term \cdot type$	application to a type
		$\beta\ \{term\}$	reflexivity of equality
		$\varsigma\ term$	symmetry of equality
		$\mu\ term\ motive^?\ \{\ case^*\ \}$	pattern match and fixpoint
		u	term variable
		$(term)$	
		$\bullet$	hole for incomplete term
$vararg$	::=	u	normal constructor argument
		$\text{-}\ u$	erased constructor argument
		$\cdot\ X$	type constructor argument
$class$	::=	$:\ typeOrKind$	
$motive$	::=	$@\ type$	motive for induction
$case$	::=	$\mid\ id\ arg^* \mapsto term$	pattern-matching cases

Figure 5: Annotated Terms