

Refinement Types & Function Extensionality

Work in Progress
Niki Vazou, IMDEA
IFIP WG 2.1

Refinement Types

`plus1R :: x:Int → {v:Int | x < v}`
`plus1R x = x + 1`

Refinement Types

$\text{plus1R} \quad :: \ x:\text{Int} \rightarrow \{v:\text{Int} \mid x < v\}$
 $\text{plus1R } x = x + 1$

$$\frac{\begin{array}{c} \dots, x:\text{Int}, \ v:\text{Int}; \ v = x + 1 \vdash x < v \\ \vdots \end{array}}{\Gamma, x:\text{Int} \vdash x + 1 :: \{v:\text{Int} \mid x < v\}} \\ \hline \Gamma \vdash \lambda x. x + 1 :: x:\text{Int} \rightarrow \{v:\text{Int} \mid x < v\}$$

Refinement Types

$\text{plus1R} \quad :: \quad x:\text{Int} \rightarrow \{v:\text{Int} \mid x < v\}$
 $\text{plus1R } x = x + 1$

$$\frac{\dots, x:\text{Int}, v:\text{Int}; v = x + 1 \vdash x < v}{\vdots}$$

$$\Gamma, x:\text{Int} \vdash x + 1 :: \{v:\text{Int} \mid x < v\}$$

$$\Gamma \vdash \lambda x. x + 1 :: x:\text{Int} \rightarrow \{v:\text{Int} \mid x < v\}$$

Γ

$e :: t$

Refinement Types

plus1R :: x:Int → {v:Int | x < v}
plus1R x = x + 1

$$\frac{\begin{array}{c} \Sigma \qquad \Phi \qquad p \\ \dots, x:\text{Int}, \quad v:\text{Int}; \quad v = x + 1 \vdash x < v \end{array}}{\vdots}$$

$\Gamma \vdash e :: t$

Refinement Typing

Refinement Types

`plus1R :: x:Int → {v:Int | x < v}`
`plus1R x = x + 1`

$\Sigma; \Phi \vdash p$

SMT

$\Gamma \vdash e :: t$

Refinement Typing

Function Equality & Refinement Types

```
plus1L x = 1 + x  
plus1R x = x + 1
```

We want to show that `plus1R == plus1L`

```
lemma :: x:Int → { plus1L x == plus1R x }  
lemma x = ()
```

Function Equality & Refinement Types

`plus1L x = 1 + x`
`plus1R x = x + 1`

We want to show that `plus1R == plus1L`

`..., x: Int; ∅ ⊢ plus1R x == plus1L x`

`Γ, x: Int ⊢ () :: {plus1R x == plus1L x}`

`Γ ⊢ λx. () :: x: Int → {plus1R x == plus1L x}`

Function Equality & Refinement Types

`plus1L x = 1 + x`
`plus1R x = x + 1`

Add an intermediate level that unfolds functions!

`..., x: Int; $\emptyset \vdash \text{plus1R } x == \text{plus1L } x$` ✗

$\Gamma, x: \text{Int} \vdash () :: \{\text{plus1R } x == \text{plus1L } x\}$

$\Gamma \vdash \lambda x. () :: x: \text{Int} \rightarrow \{\text{plus1R } x == \text{plus1L } x\}$

Function Equality & Refinement Types

$$F = \begin{cases} \text{plus1L} & x = 1 + x \\ \text{plus1R} & x = x + 1 \end{cases} \quad \phi =$$

Add an intermediate level that unfolds functions!

$$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x ; \phi$$

Function Equality & Refinement Types

$$F = \begin{cases} \text{plus1L } x = 1 + x \\ \text{plus1R } x = x + 1 \end{cases} \quad \phi = \text{plus1R } x = x + 1$$

Add an intermediate level that unfolds functions!

$$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x ; \phi$$

Function Equality & Refinement Types

$$F = \begin{cases} \text{plus1L } x = 1 + x \\ \text{plus1R } x = x + 1 \end{cases} \quad \phi = \begin{cases} \text{plus1R } x = x + 1 \\ \text{plus1L } x = 1 + x \end{cases}$$

Add an intermediate level that unfolds functions!

$$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x; \phi$$

Function Equality & Refinement Types

$$F = \begin{cases} \text{plus1L } x = 1 + x \\ \text{plus1R } x = x + 1 \end{cases} \quad \phi = \begin{cases} \text{plus1R } x = x + 1 \\ \text{plus1L } x = 1 + x \end{cases}$$

$$\underline{\dots, x:\text{Int}; \phi \vdash \text{plus1R } x == \text{plus1L } x} \quad \checkmark$$

$$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x ; \phi$$

$$\Gamma, x:\text{Int} \vdash () :: \{\text{plus1R } x == \text{plus1L } x\}$$

$$\Gamma \vdash \lambda x. () :: x:\text{Int} \rightarrow \{\text{plus1R } x == \text{plus1L } x\}$$

Function Equality & Refinement Types

$$F = \begin{cases} \text{plus1L } x = 1 + x \\ \text{plus1R } x = x + 1 \end{cases} \quad \Phi = \begin{cases} \text{plus1R } x = x + 1 \\ \text{plus1L } x = 1 + x \end{cases}$$

$\Sigma; \Phi \vdash p$

SMT

$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x ; \Phi$

$\Gamma \vdash e :: t$

Refinement Typing

Proof By Logical Evaluation (PLE)

Unfolds functions to strengthen SMT environment

- ✓ terminating
- ✓ complete
- ✓ sound

$\Sigma; \Phi \vdash p$

SMT

$F; \Sigma; \Phi \vdash p; \Phi$

PLE

$\Gamma \vdash e :: t$

Refinement Typing

Proof By Logical Evaluation (PLE)

`plus1L x = 1 + x`
`plus1R x = x + 1`

We can now show that `plus1R == plus1L`

`lemma :: x:Int → { plus1L x == plus1R x }`
`lemma x = ()`

How about:

`thm :: { plus1L == plus1R }`
`thm = ()`

Proof By Logical Evaluation (PLE)

plus1L x = 1 + x
plus1R x = x + 1

thm :: { plus1L == plus1R }
thm = ()

$\Gamma \vdash () :: \{ \text{plus1R} == \text{plus1L} \}$

Proof By Logical Evaluation (PLE)

$$F = \begin{cases} \text{plus1L} & x = 1 + x \\ \text{plus1R} & x = x + 1 \end{cases} \quad \Phi = \emptyset$$

$$F; \dots; \emptyset \vdash \text{plus1R} == \text{plus1L}; \Phi$$

$$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$$

Proof By Logical Evaluation (PLE)

$$F = \begin{cases} \text{plus1L } x = 1 + x \\ \text{plus1R } x = x + 1 \end{cases} \quad \Phi = \emptyset$$

$$\dots; \emptyset \vdash \text{plus1R} == \text{plus1L}$$

$$F; \dots; \emptyset \vdash \text{plus1R} == \text{plus1L}; \Phi$$

$$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$$

Proof By Logical Evaluation (PLE) fails on unsaturated functions

$$F = \begin{cases} \text{plus1L} & x = 1 + x \\ \text{plus1R} & x = x + 1 \end{cases} \quad \Phi = \emptyset$$

$$\dots; \emptyset \vdash \text{plus1R} == \text{plus1L}$$


✗

$$F; \dots; \emptyset \vdash \text{plus1R} == \text{plus1L}; \Phi$$

$$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$$


Proof By Logical Evaluation (PLE) fails on unsaturated functions

```
lemma :: x:Int → { plus1L x == plus1R x }  
lemma x = ()
```



Link: Function Extensionality

```
thm :: { plus1L == plus1R }  
thm = ()
```



Refinement Types & Function Extensionality

Approach I: Use extensionality as an axiom

Extensionality as an Axiom

extensionality :: $f:(a \rightarrow b) \rightarrow g:(a \rightarrow b)$
 $\rightarrow (x:a \rightarrow \{f\ x == g\ x\})$
 $\rightarrow \{f == g\}$

Extensionality as an Axiom

```
extensionality :: f:(a → b) → g:(a → b)
               → (x:a → {f x == g x})
               → {f == g}
```

```
lemma :: x:Int -> { plus1L x == plus1R x }
lemma x = ()
```

```
thm :: { plus1L == plus1R }
thm = extensionality plus1L plus1R lemma ✓
```


Refinement Types & Function Extensionality

Approach I: Use extensionality as an axiom

+ the standard route (Coq, Agda, Dafny, F^*)

+ no system modification

- requires explicit FO proofs

- it can be unsound!!!

Unsoundness of Function Extensionality

extensionality :: forall a b.

 f:(a → b) → g:(a → b)

→ (x:a → {f x == g x})

→ {f == g}

thm :: { plus1L == plus1R }

thm = extensionality @{v:IntId} @{v:IntIr}
 plus1L plus1R lemma

Unsoundness of Function Extensionality

extensionality @{v:IntId} @{v:IntIr} ::
 f:({v:IntId} → {v:IntIr})
→ g:({v:IntId} → {v:IntIr})
→ (x:{v:IntId} → {f x == g x})
→ {f == g}

Unsoundness of Function Extensionality

extensionality @{v:IntId} @{v:IntIr} ::
 f:({v:IntId} → {v:IntIr})
→ g:({v:IntId} → {v:IntIr})
→ (x:{v:IntId} → {f x == g x})
→ {f == g}

plus1L :: {v:IntId} → {v:IntIr}

Unsoundness of Function Extensionality

$$\begin{array}{l} \text{plus1L} :: \{v:\text{Int} \mid \text{domain plus1L}\} \\ \quad \rightarrow \{v:\text{Int} \mid \text{range plus1L}\} \end{array}$$

$$\text{plus1L} :: \{v:\text{Int} \mid d\} \rightarrow \{v:\text{Int} \mid r\}$$

Functional Subtyping

$$\frac{\Gamma \vdash t'_x <: t_x \quad \Gamma, x:t'_x \vdash t <: t'}{\Gamma \vdash x:t_x \rightarrow t <: x:t'_x \rightarrow t'} \text{ TSub}$$

$$\frac{\text{plus1L} :: \{v:\text{Int} \mid \text{domain plus1L}\} \rightarrow \{v:\text{Int} \mid \text{range plus1L}\}}{\text{plus1L} :: \{v:\text{Int} \mid d\} \rightarrow \{v:\text{Int} \mid r\}} \text{ TSub}$$

Functional Subtyping

$$\frac{\Gamma \vdash t'_x <: t_x \quad \Gamma, x:t'_x \vdash t <: t'}{\Gamma \vdash x:t_x \rightarrow t <: x:t'_x \rightarrow t'} \text{ TSub}$$

$$\frac{\begin{array}{l} d \Rightarrow \text{domain plus1L} \quad d \wedge \text{range plus1L} \Rightarrow r \\ \text{plus1L} :: \{v:\text{Int} \mid \text{domain plus1L}\} \\ \quad \rightarrow \{v:\text{Int} \mid \text{range plus1L}\} \end{array}}{\text{plus1L} :: \{v:\text{Int} \mid d\} \rightarrow \{v:\text{Int} \mid r\}} \text{ TSub}$$

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}
plus1L ::
g:({v:Int|d} → {v:Int|r})
→ (x:{v:Int|d} → {plus1L x == g x})
→ {plus1L == g}

$d \Rightarrow \text{domain plus1L}$

$d \wedge \text{range plus1L} \Rightarrow r$

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}
plus1L ::
g:({v:Int|d} → {v:Int|r})
→ (x:{v:Int|d} → {plus1L x == g x})
→ {plus1L == g}

$d \Rightarrow \text{domain plus1L}$

$d \wedge \text{range plus1L} \Rightarrow r$

$d \Rightarrow \text{domain plus1R}$

$d \wedge \text{range plus1R} \Rightarrow r$

plus1R :: {v:Int|d} → {v:Int|r}

Unsoundness of Function Extensionality

```
extensionality @{v:Int|d} @{v:Int|r}  
  plus1L  
  plus1R ::  
    (x:{v:Int|d} → {plus1L x == plus1R x}  
    → {plus1L == plus1R})
```

$d \Rightarrow \text{domain plus1L}$
 $d \Rightarrow \text{domain plus1R}$

$d \wedge \text{range plus1L} \Rightarrow r$
 $d \wedge \text{range plus1R} \Rightarrow r$

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}
 plus1L
 plus1R ::
 $(x:\{v:\text{Int}|d\} \rightarrow \{\text{plus1L } x == \text{plus1R } x\})$
 $\rightarrow \{\text{plus1L} == \text{plus1R}\}$

$d \Rightarrow \text{domain plus1L}$

$d \Rightarrow \text{domain plus1R}$

$d \Rightarrow \text{domain lemma}$

$d \wedge \text{range plus1L} \Rightarrow r$

$d \wedge \text{range plus1R} \Rightarrow r$

$d \wedge \text{range lemma} \Rightarrow$
 $\text{plus1L } x == \text{plus1R } x$

lemma :: $x:\{v:\text{Int}|d\} \rightarrow \{\text{plus1L } x == \text{plus1R } x\}$

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}

plus1L plus1R lemma :: {plus1L == plus1R}

$d \Rightarrow \text{domain plus1L}$

$d \Rightarrow \text{domain plus1R}$

$d \Rightarrow \text{domain lemma}$

$d \wedge \text{range plus1L} \Rightarrow r$

$d \wedge \text{range plus1R} \Rightarrow r$

$d \wedge \text{range lemma} \Rightarrow$
 $\text{plus1L } x == \text{plus1R } x$

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}

plus1L plus1R lemma :: {plus1L == plus1R}

$d \Rightarrow \text{domain plus1L}$

$d \Rightarrow \text{domain plus1R}$

$d \Rightarrow \text{domain lemma}$

$d \wedge \text{range plus1L} \Rightarrow r$

$d \wedge \text{range plus1R} \Rightarrow r$

$d \wedge \text{range lemma} \Rightarrow$
 $\text{plus1L } x == \text{plus1R } x$

Note: d only appears in assumptions!

So, if $d := \text{false}$ extensionality proofs **ANYTHING!**

Unsoundness of Function Extensionality

extensionality @{v:Int|d} @{v:Int|r}

plus1L plus1R lemma :: {plus1L == plus2 }

$d \Rightarrow \text{domain plus1L}$

$d \Rightarrow \text{domain plus1R}$

$d \Rightarrow \text{domain lemma}$

$d \wedge \text{range plus1L} \Rightarrow r$

$d \wedge \text{range plus1R} \Rightarrow r$

$d \wedge \text{range lemma} \Rightarrow$

$\text{plus1L } x == \text{plus2 } x$

Note: d only appears in assumptions!

So, if $d := \text{false}$ extensionality proofs **ANYTHING**!

Cure: default d to true

Cure of Unsoundness of Function Extensionality

extensionality $:: f:(a \rightarrow b) \rightarrow g:(a \rightarrow b)$
 $\rightarrow (x:a \rightarrow \{f\ x == g\ x\})$
 $\rightarrow \{f == g\}$

$a := \{v:\text{Int} \mid d\}$

Note: d only appears in assumptions!

because a only appears positively

Cure: default d to true

Cure of Unsoundness of Function Extensionality

extensionality :: $f:(a \rightarrow b) \rightarrow g:(a \rightarrow b)$
 $\rightarrow (x:a \rightarrow \{f\ x == g\ x\})$
 $\rightarrow \{f\ @(a \rightarrow b) == g\ @(a \rightarrow$
 $b)\}$

$a := \{v:\text{Int} \mid \text{true}\}$

Note: d only appears in assumptions!

because a only appears positively

Cure: instantiate positive variables with true

Refinement Types & Function Extensionality

Approach I: Use extensionality as an axiom

- + the standard route (Coq, Agda, Dafny, F^{*})
 - + no system modification

- requires explicit FO proofs
 - it can be unsound!!!

Refinement Types & Function Extensionality

Approach I: Use extensionality as an axiom

Approach II: Define an extensionality “layer”

The Function Extensionality Layer

thm :: { plus1L == plus1R }
thm = ()

...; $\emptyset \vdash \text{plus1R} == \text{plus2L}$



F; ...; $\emptyset \vdash \text{plus1R} == \text{plus2L}; \emptyset$

$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$

The Function Extensionality Layer

```
thm :: { plus1L == plus1R }  
thm = ()
```

SMT

PLE

$\Gamma \vdash () :: \{ \text{plus1R} == \text{plus1L} \}$

The Function Extensionality Layer

```
thm :: { plus1L == plus1R }  
thm = ()
```

SMT

PLE

Function Extensionality

$\Gamma \vdash () :: \{ \text{plus1R} == \text{plus1L} \}$

The Function Extensionality Layer

thm :: { plus1L == plus1R }
thm = ()

SMT

PLE

$$\frac{\dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x}{\dots; \emptyset \vdash \text{plus1R} == \text{plus1L}}$$
$$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$$

The Function Extensionality Layer

thm :: { plus1L == plus1R }
thm = () $\phi = \begin{matrix} \text{plus1R } x = x + 1 \\ \text{plus1L } x = 1 + x \end{matrix}$

SMT

$F; \dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x; \phi$

$$\frac{\dots, x:\text{Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x}{\dots; \emptyset \vdash \text{plus1R} == \text{plus1L}}$$

$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$

The Function Extensionality Layer

thm :: { plus1L == plus1R }
thm = () $\phi = \begin{matrix} \text{plus1R } x = x + 1 \\ \text{plus1L } x = 1 + x \end{matrix}$

...,x:Int; $\phi \vdash \text{plus1R } x == \text{plus2L } x$ ✓


F;...,x:Int; $\emptyset \vdash \text{plus1R } x == \text{plus1L } x; \phi$

$$\frac{\text{...,x:Int}; \emptyset \vdash \text{plus1R } x == \text{plus1L } x}{\text{...}; \emptyset \vdash \text{plus1R} == \text{plus1L}}$$

$\Gamma \vdash () :: \{\text{plus1R} == \text{plus1L}\}$

The Function Extensionality Layer

```
thm :: { plus1L == plus1R }  
thm = ()
```



SMT

PLE

Function Extensionality

Refinement Typing

The Function Extensionality Layer

$$\frac{\Sigma, x:a; \Phi \vdash f \ x == g \ x \quad \Sigma \vdash f, g :: a \rightarrow b}{\Sigma; \Phi \vdash f == g} =R$$

How to “uncover” equalities?

How to “uncover” equalities?

thm :: f:(a → b) → g:(a → b) → { id f == id g ⇒ f == g }
 thm = ()

$$\begin{array}{c}
 \frac{\Sigma, x:a; \Phi, \text{id } f \ x == \text{id } g \ x \vdash f \ x == g \ x}{\Sigma, x:a; \Phi, \text{id } f == \text{id } g \vdash f \ x == g \ x} =L \\
 \frac{\Sigma, x:a; \Phi, \text{id } f == \text{id } g \vdash f \ x == g \ x}{\Sigma; \Phi, \text{id } f == \text{id } g \vdash f == g} =R \\
 \frac{\Sigma; \Phi, \text{id } f == \text{id } g \vdash f == g}{\Sigma; \Phi \vdash \text{id } f == \text{id } g \Rightarrow f == g} \Rightarrow R
 \end{array}$$

The Function Extensionality Layer

Algorithm

Step I: \Rightarrow , \wedge normalization

Step II: $=R$

Step III: $=L$

The Function Extensionality Layer

Algorithm

Step I: \Rightarrow , \wedge normalization

Step II: $=_R$

Step III: $=_L$

Properties

✓ Soundness

✓ Termination

✗ Completeness

Incompleteness in Presence of ADTs

thm :: $f:(a \rightarrow b) \rightarrow g:(a \rightarrow b) \rightarrow \{\text{first } f == \text{first } g \Rightarrow f == g\}$
 $\text{first } f \ (x,y) = (f \ x,y)$

!!no pair available!!

$$\Sigma, x:a; \Phi, \text{first } f == \text{first } g \vdash f \ x == g \ x$$

$$\text{=R}$$
$$\Sigma; \Phi, \text{first } f == \text{first } g \vdash f == g$$

$$\Sigma; \Phi \vdash \text{first } f == \text{first } g \Rightarrow f == g$$
$$\Rightarrow \text{R}$$

Incompleteness in Presence of ADTs

thm :: f:(a → b) → g:(a → b) → {first f == first g ⇒ f == g}
first f (x,y) = (f x,y)

$$\frac{\Sigma, x:a, y:b; \Phi, \text{first } f (x,y) == \text{first } g (x,y) \quad \vdash f x == g x}{\text{}} =L+$$

$$\frac{\Sigma, x:a; \Phi, \text{first } f == \text{first } g \vdash f x == g x}{\text{}} =R$$

$$\frac{\Sigma; \Phi, \text{first } f == \text{first } g \vdash f == g}{\text{}} \Rightarrow R$$
$$\Sigma; \Phi \vdash \text{first } f == \text{first } g \Rightarrow f == g$$

Function Extensionality + ADTs

Soundness
(all types has inhabitant)

Termination
(no recursive ADTS)

? Completeness

Refinement Types & Function Extensionality

PLE automates **saturated** function equality

For saturation we need extensionality

Approach I: Use extensionality as an axiom

- requires explicit FO proofs
- can be unsound

Approach II: Define an extensionality “layer”

- + smooth interaction with PLE
- + increases automation
- ? completeness

Thanks!