

Gradual Liquid Types

Niki Vazou^{*}, Éric Tanter⁺, and David Van Horn^{*}
^{*} University of Maryland ⁺ University of Chile

Liquid Types

fast type checking & inference for

Refinement Types

Refinement Types

$(/)$:: $\text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

Basic Type

Refinement

Refinement Types

$$(\text{ } / \text{ }) :: \text{Int} \rightarrow \{v \mid 0 < v\} \rightarrow \text{Int}$$

Simplification

Refinement Types

```
(/)      :: Int -> {v | 0 < v} -> Int  
isPos    :: Int -> Bool
```

```
divIf    :: Int -> Int  
divIf x = if isPos x then 1/x else 1/(1-x)
```

Q: What is a refinement type for `divIf`?

A: Let's ask Liquid Inference!

Liquid Types

```
(/)      :: Int -> {v | 0 < v} -> Int  
isPos    :: Int -> Bool
```

```
divIf    :: x:{Int | false} -> {Int | false}  
divIf x = if isPos x then 1/x else 1/(1-x)
```

Problem: Inferred type for `divIf` insensible!

Liquid Types

```
divIf :: x:{Int | 0 < x} -> {Int | true}
divIf x = if isPos x then 1/x else 1/(1-x)
```

```
client = divIf 42
```

Problem: Inferred type for `divIf` insensible!

Worse: Inferred type is non-modular!

Liquid Types

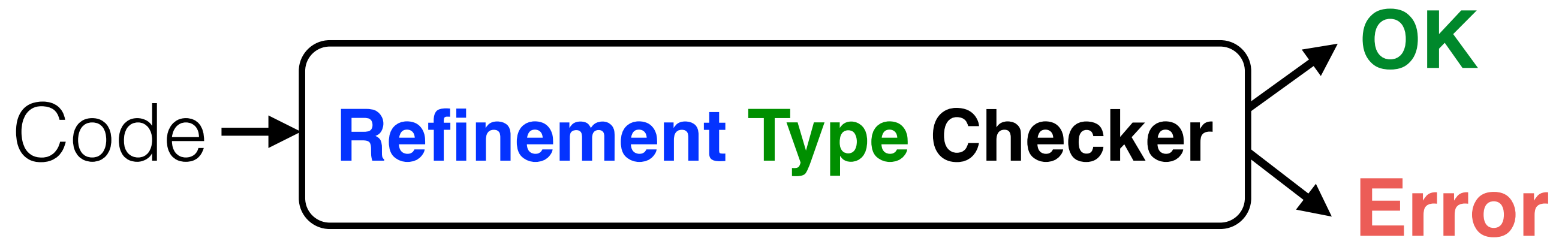
Problem: To understand errors ...
you need to know how inference works!

Liquid Types

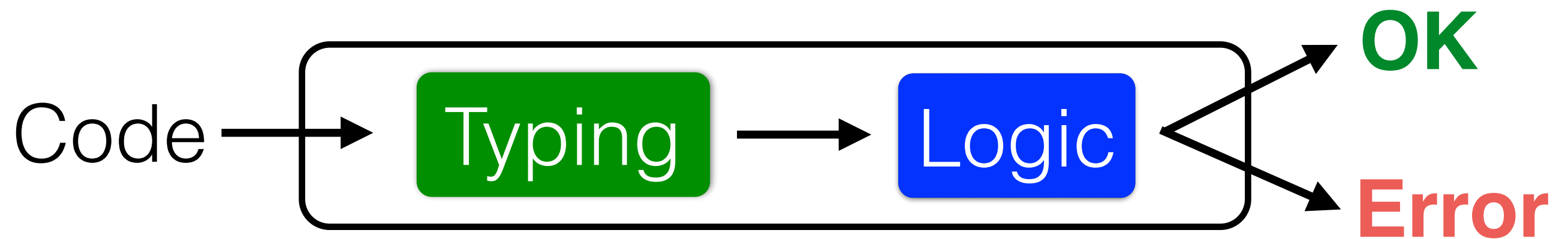
Problem: To understand errors ...
you need to know how inference works!

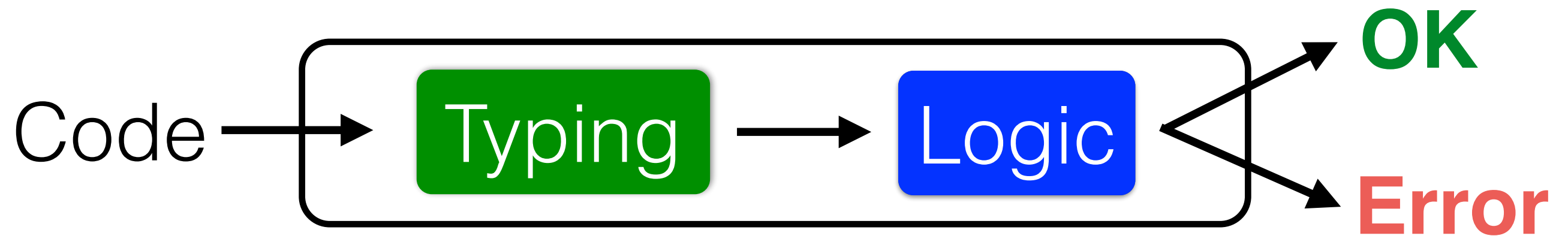
Let's start by how refinement types work!

Refinement Types



Refinement Types





```
(/) :: Int -> {v:Int | 0 < v} -> Int
```

```
isPos :: x:Int -> {b | b <=> 0 < x}
```

```
divIf x = if isPos x then 1/x else 1/(1-x)
```



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

isPos $:: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \mid - \{v \mid v = x\} \quad <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \mid - \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } \underline{1/x} \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \vdash \underline{\{v \mid v = x\}} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } \underline{x} = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$\underline{x}:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \underline{b}\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid \underline{b \Leftrightarrow 0 < x}\}$

$\text{divIf } x = \text{if } \underline{\text{isPos } x} \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid \underline{b \Leftrightarrow 0 < x}, b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$ $:: \text{Int} \rightarrow \{v:\text{Int} \mid \underline{0 < v}\} \rightarrow \text{Int}$

$\text{isPos} :: x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \vdash \{v \mid v = x\} <: \{v \mid \underline{0 < v}\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$



$(/)$:: $\text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

isPos :: $x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \vdash \{v \mid v = 1-x\} <: \{v \mid 0 < v\}$

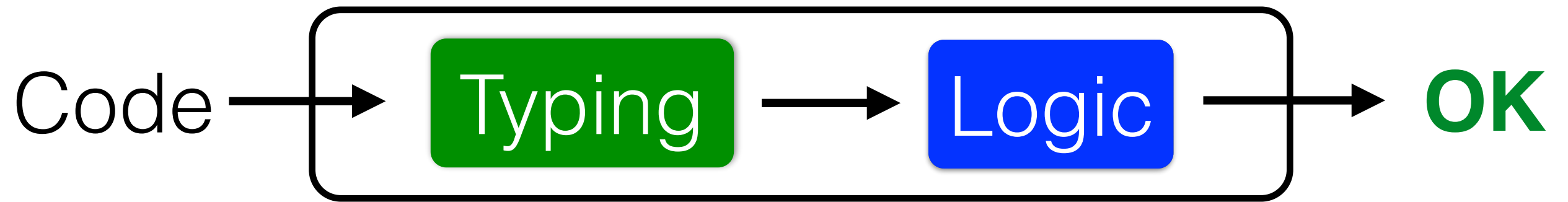


$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, b\} \mid - \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid b \Leftrightarrow 0 < x, \neg b\} \mid - \{v \mid v = 1 - x\} <: \{v \mid 0 < v\}$

$\text{true}, \quad b \Leftrightarrow 0 < x, b \quad \Rightarrow \quad v = x \quad \Rightarrow \quad 0 < v$

$\text{true}, \quad b \Leftrightarrow 0 < x, \neg b \quad \Rightarrow \quad v = 1 - x \quad \Rightarrow \quad 0 < v$



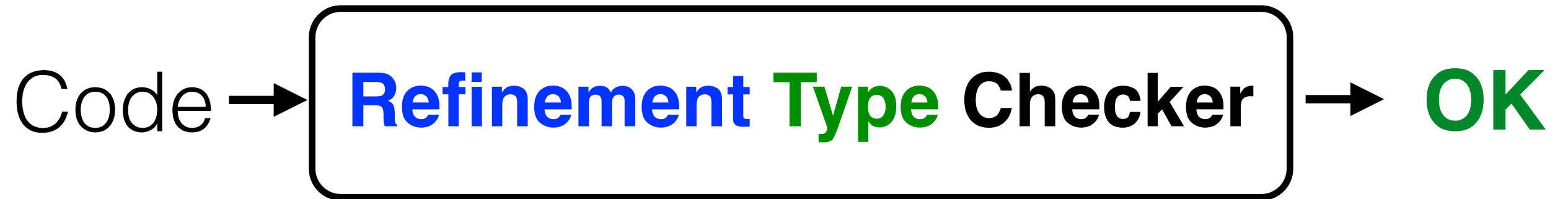
true,	$b \Leftrightarrow 0 < x, b$	\Rightarrow	$v = x$	\Rightarrow	$0 < v$
true,	$b \Leftrightarrow 0 < x, \neg b$	\Rightarrow	$v = 1 - x$	\Rightarrow	$0 < v$

Code → **Refinement Type Checker** → **OK**

$(/)$:: $\text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

isPos :: $x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$



$(/)$:: $\text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

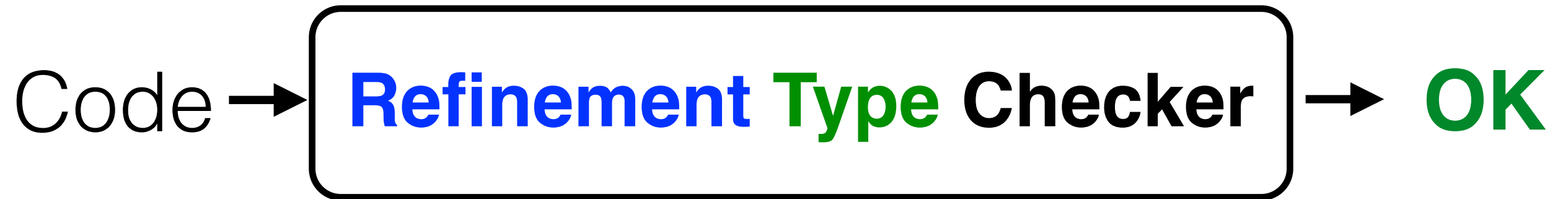
isPos :: $x:\text{Int} \rightarrow \{b \mid b \Leftrightarrow 0 < x\}$

$\text{divIf } x = \text{if } \text{isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\{0 < x\}$

$x:\{x \leq 0\}$

What if isPos is not verified?



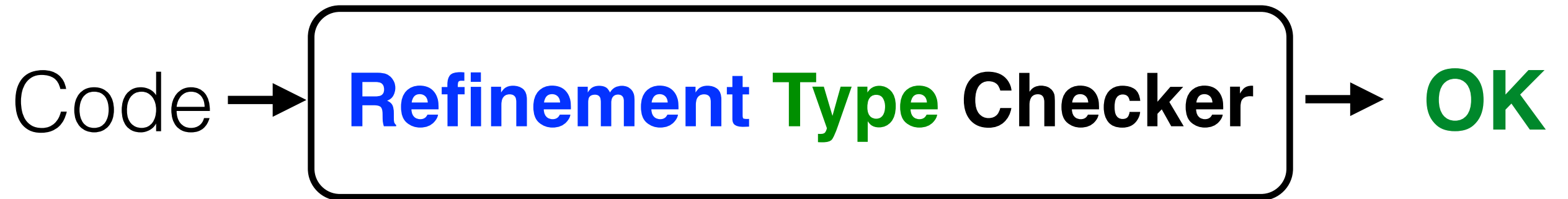
$(\text{ } / \text{ }) :: \text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x : \{0 < x\}$

$x : \{x \leq 0\}$

What if **isPos** is not verified?



$(/)$:: $\text{Int} \rightarrow \{v:\text{Int} \mid 0 < v\} \rightarrow \text{Int}$

divIf :: $x:\{\text{Int} \mid ?\} \rightarrow \text{Int}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\{0 < x\}$

$x:\{x \leq 0\}$

What if isPos is not verified?

Is there a type for x that makes divIf OK?

Is there a type for x that makes `divIf` OK?

```
divIf :: x:{Int | ?} -> Int  
divIf x = if isPos x then 1/x else 1/(1-x)
```

$x : \{0 < x\}$

$x : \{x \leq 0\}$

For every occurrence of x ,
there exists a predicate p , so that

$x : \{\text{Int} \mid p \ x\}$

Gradual Refinement Types

$$x : \{ \text{Int} \mid ? \}$$

For every occurrence of x ,
there exists a predicate p , so that

$$x : \{ \text{Int} \mid p \ x \}$$



```
divIf :: x:{Int | ? } -> Int  
divIf x = if isPos x then 1/x else 1/(1-x)
```

Code

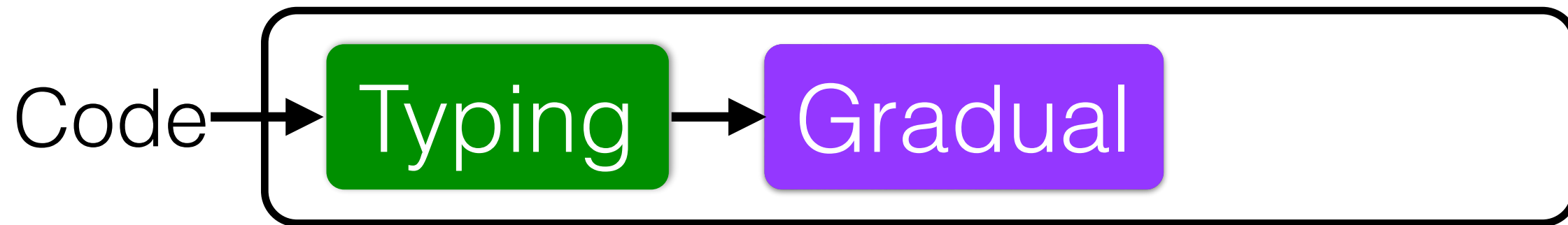
→ Typing

$\text{divIf} :: x : \{\text{Int} \mid ?\} \rightarrow \text{Int}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

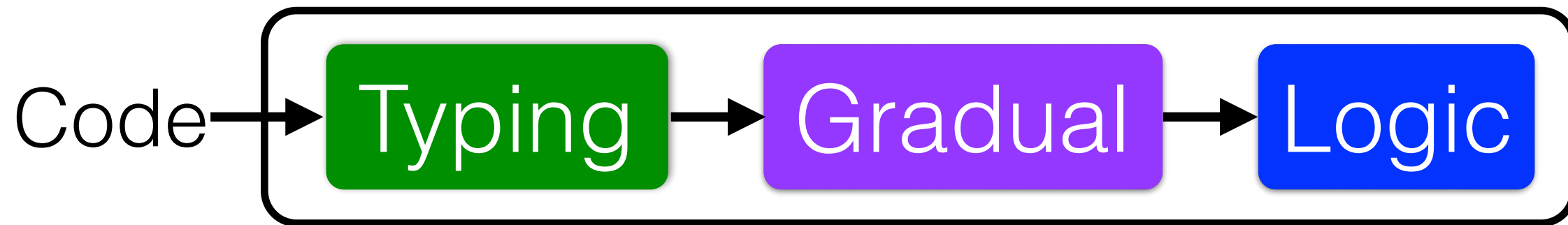
$x : \{x \mid ?\}, b : \{b \mid b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\}$

$x : \{x \mid ?\}, b : \{b \mid \neg b\} \vdash \{v \mid v = 1 - x\} <: \{v \mid 0 < v\}$



$$\begin{aligned}
 &x:\{x \mid ?\}, b:\{b \mid b\} \vdash \{v \mid v = x\} <: \{v \mid 0 < v\} \\
 &x:\{x \mid ?\}, b:\{b \mid \neg b\} \vdash \{v \mid v = 1 - x\} <: \{v \mid 0 < v\}
 \end{aligned}$$

$$\begin{aligned}
 \exists p. p \ x, \quad b &\quad \Rightarrow \quad v = x \quad \Rightarrow \quad 0 < v \\
 \exists p. p \ x, \quad \neg b &\quad \Rightarrow \quad v = 1 - x \quad \Rightarrow \quad 0 < v
 \end{aligned}$$



$$\begin{array}{lll}
 \exists p. \text{ p } x, \text{ b} & \Rightarrow & v = x \Rightarrow 0 < v \\
 \exists p. \text{ p } x, \neg \text{ b} & \Rightarrow & v = 1 - x \Rightarrow 0 < v
 \end{array}$$

$$\begin{array}{lll}
 0 < x, \text{ b} & \Rightarrow & v = x \Rightarrow 0 < v \\
 x \leq 0, \neg \text{ b} & \Rightarrow & v = 1 - x \Rightarrow 0 < v
 \end{array}$$

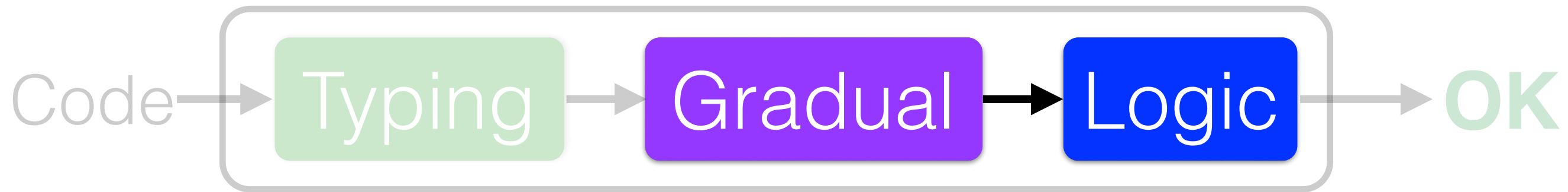


$$\begin{array}{llll} 0 < x, & b & \Rightarrow & v = x \Rightarrow 0 < v \\ x \leq 0, & \neg b & \Rightarrow & v = 1 - x \Rightarrow 0 < v \end{array}$$



```
divIf :: x:{Int | ? } -> Int
divIf x = if isPos x then 1/x else 1/(1-x)
```

$x : \{0 < x\}$ $x : \{x \leq 0\}$



```
divIf :: x:{Int | ? } -> Int
divIf x = if isPos x then 1/x else 1/(1-x)
```

$x : \{0 < x\}$ $x : \{x \leq 0\}$

Two curved arrows point from the type annotations below to the corresponding expressions in the code above. The first arrow points from $x : \{0 < x\}$ to the $1/x$ expression. The second arrow points from $x : \{x \leq 0\}$ to the $1/(1-x)$ expression.

How do we solve existential over predicates?

How do we solve existential over predicates?

Gradual Refinement Types

In Theory *

How do we solve existential over predicates?

In Practise

*Lehmann & Tanter [POPL'16]

How do we solve existential over predicates?


Problem: Domain of predicates is infinite

$$p \in \{ 0 < x, x \leq 0, b \Leftrightarrow 0 < x, b, \neg b, \dots \}$$

How do we solve existential over predicates?

Problem: Domain of predicates is infinite

Solution: Predicates over finite domain

$$p \in \{ 0 < x, x \leq 0, b \Leftrightarrow 0 < x, b, \neg b \}$$


Predicates over finite domain

Liquid Types

Application: Type Inference

Liquid Types

```
isPos :: x:Int -> {b:Bool | p b }
```

```
isPos x = 0 < x
```

```
divIf x = if isPos x then 1/x else 1/(1-x)
```

Solution for **p** so that **divIf** is OK?

Liquid Types

`isPos :: x:Int -> {b:Bool | p b }`

`isPos x = 0 < x`

`divIf x = if isPos x then 1/x else 1/(1-x)`

`x:Int` $\vdash \{v \mid v = 0 < x\}$ $<: \{v \mid p \ v\}$

`x:Int, b:{b | p b, b}` $\vdash \{v \mid v = x\}$ $<: \{v \mid 0 < v\}$

`x:Int, b:{b | p b, ¬b}` $\vdash \{v \mid v = 1-x\}$ $<: \{v \mid 0 < v\}$

Liquid Types

$x:\text{Int} \quad \quad \quad \vdash \{v \mid v=0 < x\} \quad <: \{v \mid p \ v\}$
 $x:\text{Int}, b:\{b \mid p \ b, \ b\} \vdash \{v \mid v = x\} \quad <: \{v \mid 0 < v\}$
 $x:\text{Int}, b:\{b \mid p \ b, \ \neg b\} \vdash \{v \mid v = 1-x\} \quad <: \{v \mid 0 < v\}$

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int} \quad \quad \quad |- \quad \{v \mid v=0 < x\} \quad <: \quad \{v \mid p \ v\}$
 $x:\text{Int}, b:\{b \mid p \ b, \quad b\} \quad |- \quad \{v \mid v = x\} \quad <: \quad \{v \mid 0 < v\}$
 $x:\text{Int}, b:\{b \mid p \ b, \neg b\} \quad |- \quad \{v \mid v = 1-x\} \quad <: \quad \{v \mid 0 < v\}$

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

?

$p \ v \in \{ \boxed{0 < v}, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

?

$p \ v \in \{ \boxed{0 < v}, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \boxed{v \leq 0}, v \Leftrightarrow 0 < x, v, \neg v \}$

Liquid Types

$x:\text{Int} \quad \vdash \{v \mid v=0 < x\} \quad <: \{v \mid p \ v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ b\} \quad \vdash \{v \mid v = x\} \quad <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\} \quad \vdash \{v \mid v = 1-x\} \quad <: \{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \boxed{v \leq 0}, v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int} \quad \vdash \{v \mid v=0 < x\} \quad <: \{v \mid p \ v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ b\} \quad \vdash \{v \mid v = x\} \quad <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\} \quad \vdash \{v \mid v = 1-x\} \quad <: \{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \boxed{v}, \neg v \}$

Liquid Types

$x:\text{Int} \quad \vdash \{v \mid v=0 < x\} \quad <: \{v \mid p \ v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ b\} \quad \vdash \{v \mid v = x\} \quad <: \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\} \quad \vdash \{v \mid v = 1-x\} \quad <: \{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \boxed{v}, \neg v \}$

Liquid Types

$x:\text{Int} \quad \quad \quad |- \quad \{v \mid v=0 < x\} \quad <: \quad \{v \mid p \ v\}$

$x:\text{Int}, b:\{b \mid p \ b, \quad b\} \quad |- \quad \{v \mid v = x\} \quad <: \quad \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid p \ b, \neg b\} \quad |- \quad \{v \mid v = 1-x\} \quad <: \quad \{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \quad \boxed{\neg v} \}$

Liquid Types

$x:\text{Int} \quad \quad \quad |- \quad \{v \mid v=0 < x\} \quad <: \quad \{v \mid p \ v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ b\} \quad |- \quad \{v \mid v = x\} \quad <: \quad \{v \mid 0 < v\}$

$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\} \quad |- \quad \{v \mid v = 1-x\} \quad <: \quad \{v \mid 0 < v\}$

?

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \boxed{\neg v} \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

$p \ v \in \{ 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x, \ v, \ \neg v \}$

Liquid Types

$x:\text{Int}$	\vdash	$\{v \mid v=0 < x\}$	$<:$	$\{v \mid p \ v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ b\}$	\vdash	$\{v \mid v = x\}$	$<:$	$\{v \mid 0 < v\}$
$x:\text{Int}, b:\{b \mid p \ b, \ \neg b\}$	\vdash	$\{v \mid v = 1-x\}$	$<:$	$\{v \mid 0 < v\}$

$p \ v = \quad 0 < v, \ v \leq 0, \ v \Leftrightarrow 0 < x \ v, \ \neg v$

Liquid Types

```
isPos :: x:Int -> {b:Bool | p b }
```

```
isPos x = 0 < x
```

```
divIf x = if isPos x then 1/x else 1/(1-x)
```

$p\ v = v \Leftrightarrow 0 < x$

Liquid Types

```
isPos :: x:Int -> {b:Bool | p b }
```

```
isPos x = 0 < x
```

```
divIf x = if isPos x then 1/x else 1/(1-x)
```

$$p\ v = v \Leftrightarrow 0 < x$$

there exists a predicate p , so that
for every occurrence of b ,

$$b : \{ \text{Bool} \mid p\ b \}$$

Liquid Types

```
isPos :: x:Int -> {b:Bool | p b }
```

```
isPos x = 0 < x
```

```
divIf x = if isPos x then 1/x else 1/(1-x)
```

there exists a predicate p , so that

for every occurrence of b ,

$$b : \{ \text{Bool} \mid p \ b \}$$

Gradual Liquid Types

```
divIf :: x:{Int | ? } -> Int  
divIf x = if isPos x then 1/x else 1/(1-x)
```

there exists a predicate p , so that
for every occurrence of x ,
 $x:\{\text{Int} \mid p\ x\}$

Gradual Liquid Types

```
divIf :: x:{Int | ? } -> Int
divIf x = if isPos x then 1/x else 1/(1-x)
```

for every occurrence of x ,
there exists a predicate p , so that
 $x:\{\text{Int} \mid p\ x\}$

Gradual Liquid Types

$\text{divIf} :: x:\{\text{Int} \mid ?\} \rightarrow \text{Int}$

$\text{divIf } x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x:\{x \mid ?\}, b:\{b \mid b\} \vdash \{v \mid v=x\} <: \{v \mid 0 < v\}$

$x:\{x \mid ?\}, b:\{b \mid \neg b\} \vdash \{v \mid v=1-x\} <: \{v \mid 0 < v\}$

Gradual Liquid Types

```
divIf :: x:{Int | ? } -> Int
divIf x = if isPos x then 1/x else 1/(1-x)
```

$$x:\{x \mid p_1\}, b:\{b \mid b\} \vdash \{v \mid v=x\} <: \{v \mid 0 < v\}$$
$$x:\{x \mid p_2\}, b:\{b \mid \neg b\} \vdash \{v \mid v=1-x\} <: \{v \mid 0 < v\}$$

Gradual Liquid Types

$$\begin{aligned} x:\{x \mid p_1\}, b:\{b \mid b\} &\vdash \{v \mid v=x\} <: \{v \mid 0 < v\} \\ x:\{x \mid p_2\}, b:\{b \mid \neg b\} &\vdash \{v \mid v=1-x\} <: \{v \mid 0 < v\} \end{aligned}$$
$$p_1 \quad v \in \{0 < v, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v\}$$
$$p_2 \quad v \in \{0 < v, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v\}$$

Gradual Liquid Types


$$\begin{aligned} x &: \{x \mid p_1\}, b : \{b \mid b\} \mid - \{v \mid v = x\} <: \{v \mid 0 < v\} \\ x &: \{x \mid p_2\}, b : \{b \mid \neg b\} \mid - \{v \mid v = 1 - x\} <: \{v \mid 0 < v\} \end{aligned}$$
$$p_1 \quad v \in \{0 < v, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v\}$$
$$p_2 \quad v \in \{0 < v, v \leq 0, v \Leftrightarrow 0 < x, v, \neg v\}$$

Gradual Liquid Types

`divIf` :: $x : \{\text{Int} \mid ?\} \rightarrow \text{Int}$

`divIf` $x = \text{if isPos } x \text{ then } 1/x \text{ else } 1/(1-x)$

$x : \{0 < x\}$



$x : \{x \leq 0\}$



For every occurrence of x ,
there exists a predicate p , so that

$x : \{\text{Int} \mid p \ x\}$

Gradual Liquid Types

In Theory

How do we solve existential over predicates?

Exhaustive search over finite domain.

In Practise

Gradual Liquid Types

Exhaustive search over finite domain.

In Practise

Advantage: Type Inference

Gradual Liquid Types

Type Inference

$$x_1 : \{ ? \}, \dots, x_n : \{ 0 < x_n \} \quad | - \quad \{ v \mid \text{true} \} <: \{ v \mid r_1 \}$$
$$y_1 : \{ r_1 \}, \dots, y_n : \{ ? \} \quad | - \quad \{ v \mid ? \} <: \{ v \mid v < 0 \}$$

For every gradual solution

If there exists a static solution

then return OK

return Error

Gradual Liquid Types

Exhaustive search over finite domain.

In Practise

Advantage: Type Inference

Gradual Liquid Types

Exhaustive search over finite domain.

In Practise

Advantage: Type Inference

Disadvantage: A lot of extra work

Gradual Liquid Types

How do we solve existential over predicates?

Exhaustive search over finite domain.

Disadvantage: A lot of extra work

Our answer: Certificate Generation

Application: Error Reporting

Application: Error Reporting

$(!!) :: xs : [a] \rightarrow \{Int \mid \boxed{?}\} \rightarrow a$

$(x : xs) !! 0 = x$

$(x : xs) !! i = xs !! (i-1)$

$_ !! _ = \text{error "Out of bounds!"}$

Application: Error Reporting

$(!!) :: xs : [a] \rightarrow \{Int \mid \boxed{?}\} \rightarrow a$

$(x : xs) !! 0 = x$

$(x : xs) !! i = xs !! (i-1)$

$_ !! _ = \text{error "Out of bounds!"}$

Q: Give me all the certificates

A: Demo

Gradual Liquid Types

Application: Error Reporting

Gradual Liquid Types

In Theory

How do we solve existential over predicates?
Exhaustive search over finite domain.

In Practise

Advantage: Type Inference

Application: Error Reporting

Thanks!