

Well-typed programs can't be blamed

Philip Wadler

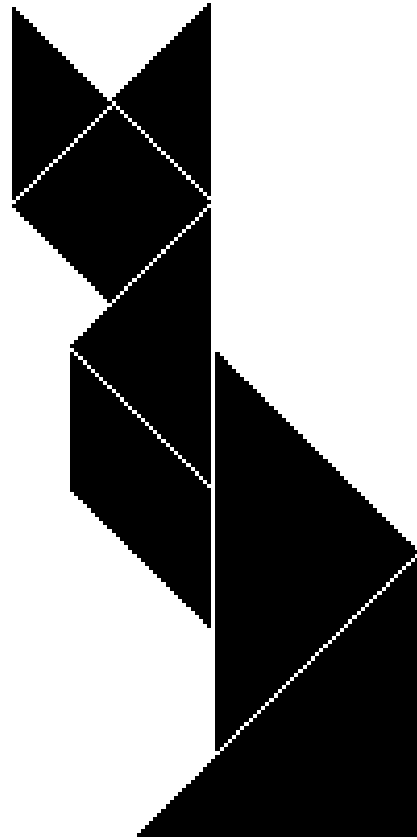
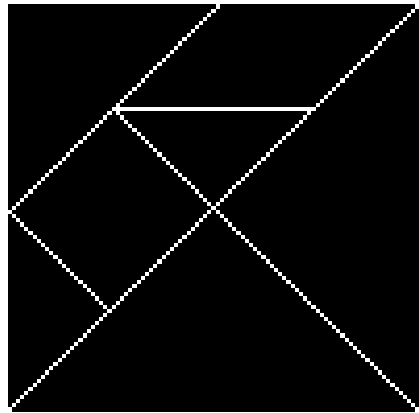
University of Edinburgh

NII Shonan Meeting

26–30 May 2014

Publications

- Wadler and Findler
Well-typed programs can't be blamed
ESOP 2009
- Siek and Wadler
Threesomes, with and without blame
POPL 2010
- Ahmed, Findler, Siek, and Wadler
Blame for all
POPL 2011
- Siek, Thiemann, Wadler
Blame, coercions, and threesomes, precisely
draft



A repeated theme

Henglein (1994):

Coercions

Findler and Felleisen (2002):

Contracts

Flanagan (2006):

Hybrid types

Siek and Taha (2006):

Gradual types

A repeated theme

Dynamic in .Net
C#, Visual Basic

JavaScript
TypeScript

Dart

Perl 6.0

Python
Reticulated Python

Part I

From untyped to typed

An untyped program

```
[let  
   $x = 2$   
   $f = \lambda y. y + 1$   
   $h = \lambda g. g (g x)$   
in  
   $h f$ ]
```

→

```
[4]
```

A typed program

let

$x = 2$

$f = \lambda y : \text{Int}. y + 1$

$h = \lambda g : \text{Int} \rightarrow \text{Int}. g (g x)$

in

$h f$

→

$4 : \text{Int}$

A partly typed program—narrowing

let

$x = 2$

$f = [\lambda y. y + 1] : \star \xRightarrow{p} \text{Int} \rightarrow \text{Int}$

$h = \lambda g : \text{Int} \rightarrow \text{Int}. g (g x)$

in

$h f$

→

$4 : \text{Int}$

A partly typed program—narrowing

```
let
  x = 2
  f = [λy. false] : ★  $\stackrel{p}{\Rightarrow}$  Int → Int
  h = λg : Int → Int. g (g x)
in
  h f
→
  blame p
```

Positive (covariant): blame the term contained in the cast

Another partly typed program—widening

let

$x = [2]$

$f = (\lambda y : \text{Int}. y + 1) : \text{Int} \rightarrow \text{Int} \xRightarrow{p} \star$

$h = [\lambda g. g (g x)]$

in

$[h f]$

→

$[4]$

Another partly typed program—widening

let

$x = [\text{true}]$

$f = (\lambda y : \text{Int}. y + 1) : \text{Int} \rightarrow \text{Int} \xrightarrow{p} \star$

$h = [\lambda g. g (g x)]$

in

$[h f]$

→

blame \bar{p}

Negative (contravariant): blame the context containing the cast

Part II

Untyped and supertyped

Untyped = Uni-typed

$$[x] = x$$

$$[k] = k : A \xrightarrow{p} \star \quad \text{if } \text{ty}(k) = A$$

$$[op(\vec{M})] = op([M] : \star \xrightarrow{\vec{p}} \vec{A}) : B \xrightarrow{p} \star \quad \text{if } \text{ty}(op) = \vec{A} \rightarrow B$$

$$[\lambda x. N] = (\lambda x : \star. [N]) : \star \rightarrow \star \Rightarrow \star$$

$$[L M] = ([L] : \star \xrightarrow{p} \star \rightarrow \star) [M]$$

(slogan due to Dana Scott, repeated by Bob Harper)

Contracts

$$\text{Nat} = \{x : \text{Int} \mid x \geq 0\}$$

let

$$x = 2 : \text{Int} \xrightarrow{p} \text{Nat}$$

$$f = (\lambda y : \text{Int}. y + 1) : \text{Int} \rightarrow \text{Int} \xrightarrow{q} \text{Nat} \rightarrow \text{Nat}$$

$$h = \lambda g : \text{Nat} \rightarrow \text{Nat}. g (g x)$$

in

$$h f$$

→

$$4 : \text{Nat}$$

Part III

The Blame Calculus

Notation

It took us four years to find the right notation!

$$\langle A \Rightarrow B \rangle^p s$$

$$\langle B \Leftarrow A \rangle^p s$$

$$s : A \xRightarrow{p} B$$

We want composition to be easy to read:

$$\langle B \Rightarrow C \rangle^q \langle A \Rightarrow B \rangle^p s$$

$$\langle C \Leftarrow B \rangle^q \langle B \Leftarrow A \rangle^p s$$

$$s : A \xRightarrow{p} B : B \xRightarrow{q} C$$

And there is a convenient abbreviation:

$$s : A \xRightarrow{p} B \xRightarrow{q} C$$

Blame calculus: Compatibility

$$\frac{\Gamma \vdash M : A \quad A \prec B}{\Gamma \vdash (M : A \xRightarrow{p} B) : B}$$

$$\overline{A \prec \star} \quad \overline{\star \prec A}$$

$$\overline{\iota \prec \iota}$$

$$\frac{A' \prec A \quad B \prec B'}{A \rightarrow B \prec A' \rightarrow B'}$$

Reductions

Ground types $G ::= \iota \mid \star \rightarrow \star$

$$\begin{aligned} (V : A \rightarrow B \xRightarrow{p} A' \rightarrow B') W &\longrightarrow (V (W : A' \xRightarrow{\bar{p}} A) : B \xRightarrow{p} B') \\ V : \iota \xRightarrow{p} \iota &\longrightarrow V \\ V : A \xRightarrow{p} \star &\longrightarrow V : A \xRightarrow{p} G \Rightarrow \star \quad \text{if } \star \neq A \prec G \\ V : G \Rightarrow \star \xRightarrow{p} A &\longrightarrow \begin{cases} V : G \xRightarrow{p} A & \text{if } G \prec A \\ \text{blame } p & \text{if } G \not\prec A \end{cases} \end{aligned}$$

Blame

$[2] : \star \xRightarrow{p} \text{Int}$

=

$2 : \text{Int} \Rightarrow \star \xRightarrow{p} \text{Int}$

→

2

$[\text{true}] : \star \xRightarrow{p} \text{Int}$

=

$\text{true} : \text{Bool} \Rightarrow \star \xRightarrow{p} \text{Int}$

→

blame p

The Blame Game—widening

$((\lambda y : \text{Int}. y + 1) : \text{Int} \rightarrow \text{Int} \xRightarrow{p} \star \rightarrow \star) [2]$

→

$(\lambda y : \text{Int}. y + 1) ([2] : \star \xRightarrow{\bar{p}} \text{Int}) : \text{Int} \xRightarrow{p} \star$

→

[3]

The Blame Game—widening

$((\lambda y : \text{Int}. y + 1) : \text{Int} \rightarrow \text{Int} \xrightarrow{p} \star \rightarrow \star) [\text{true}]$

→

$(\lambda y : \text{Int}. y + 1) ([\text{true}] : \star \xrightarrow{\bar{p}} \text{Int}) : \text{Int} \xrightarrow{p} \star$

→

$\text{blame } \bar{p}$

Widening can give rise to negative blame, but never positive blame

The Blame Game—narrowing

$$((\lambda y : \star. [y + 1]) : \star \rightarrow \star \xRightarrow{p} \text{Int} \rightarrow \text{Int}) 2$$

→

$$(\lambda y : \star. [y + 1]) (2 : \text{Int} \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} \text{Int}$$

→

3

The Blame Game—narrowing

$((\lambda y : \star. [\text{false}]) : \star \rightarrow \star \xrightarrow{p} \text{Int} \rightarrow \text{Int}) 2$

→

$(\lambda y : \star. [\text{false}]) (2 : \text{Int} \xrightarrow{\bar{p}} \star) : \star \xrightarrow{p} \text{Int}$

→

`blame p`

Narrowing can give rise to positive blame, but never negative blame

Part IV

Subtyping

$<:$

$<:^+$

$<:^-$

$<:^n$

Subtype

$$\frac{}{\star <: \star}$$

$$\frac{}{\iota <: \iota} \quad \frac{A <: G}{A <: \star}$$

$$\frac{A' <: A \quad B <: B'}{A \rightarrow B <: A' \rightarrow B'}$$

Example:

$$\frac{\frac{}{\text{Int} <: \text{Int}}}{\text{Int} <: \star} \quad \frac{\frac{}{\text{Int} <: \text{Int}}}{\text{Int} <: \star}}{\star \rightarrow \text{Int} <: \text{Int} \rightarrow \star}$$

Positive subtype—widening

$$\overline{A <:^+ \star}$$

$$\overline{l <: l}$$

$$\frac{A' <:^- A \quad B <:^+ B'}{A \rightarrow B <:^+ A' \rightarrow B'}$$

Example:

$$\frac{\overline{\star <:^- \text{Int}} \quad \overline{\text{Int} <:^+ \star}}{\text{Int} \rightarrow \text{Int} <: \star \rightarrow \star}$$

Negative subtype—narrowing

$$\overline{\star <:^- A}$$

$$\frac{}{\overline{\iota <: \iota}} \quad \frac{A <:^- G}{A <:^- \star}$$

$$\frac{A' <:^+ A \quad B <:^- B'}{A \rightarrow B <:^- A' \rightarrow B'}$$

Example:

$$\frac{\overline{\text{Int} <:^+ \star} \quad \overline{\star <:^- \text{Int}}}{\star \rightarrow \star <:^- \text{Int} \rightarrow \text{Int}}$$

Naive subtype

$$\frac{}{A <:_n \star}$$

$$\frac{}{l <:_n l}$$

$$\frac{A <:_n A' \quad B <:_n B'}{A \rightarrow B <:_n A' \rightarrow B'}$$

Example:

$$\frac{\frac{}{\text{Int} <:_n \star} \quad \frac{}{\text{Int} <:_n \star}}{\text{Int} \rightarrow \text{Int} <:_n \star \rightarrow \star}$$

Part V

The Blame Theorem

Blame Safety

$$\frac{}{x \text{ safe } p} \quad \frac{N \text{ safe } p}{\lambda x. N \text{ safe } p} \quad \frac{L \text{ safe } p \quad M \text{ safe } p}{LM \text{ safe } p}$$

$$\frac{M \text{ safe } p \quad A <:^+ B}{M : A \xRightarrow{p} B \text{ safe } p}$$

$$\frac{M \text{ safe } p \quad A <:^- B}{M : A \xRightarrow{\bar{p}} B \text{ safe } p}$$

$$\frac{s \text{ safe } p \quad p \neq q \quad \bar{p} \neq q}{M : A \xRightarrow{q} B \text{ safe } p}$$

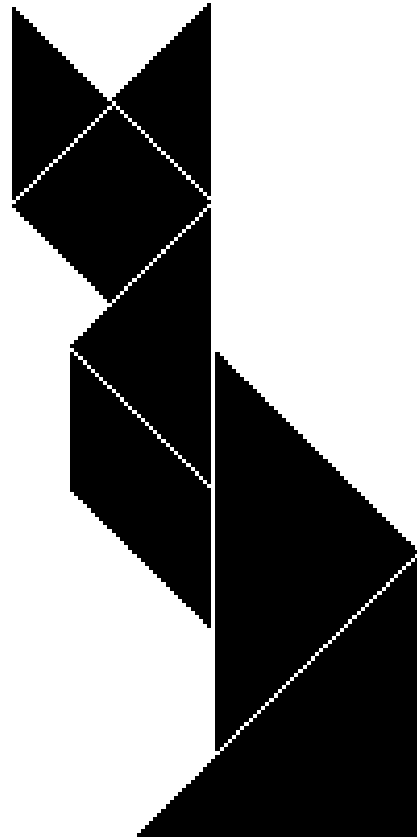
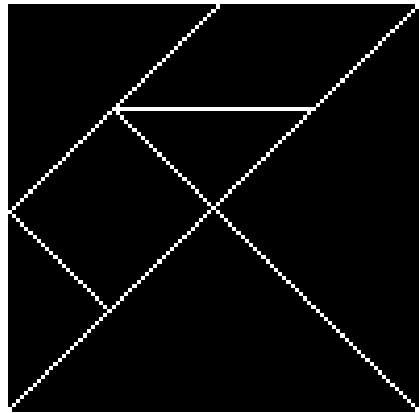
The Blame Theorem

Preservation

If M safe p and $M \longrightarrow N$ then N safe p .

Progress

If M safe p then $M \not\rightarrow$ blame p .



The First Tangram Theorem

$A <: B$ if and only if $A <:^+ B$ and $A <:^- B$

The First Blame Corollary

Let M be a term where $N : A \xrightarrow{p} B$ is the only subterm with label p .

If $A <: B$ then $M \not\rightarrow \text{blame } p$ and $M \not\rightarrow \text{blame } \bar{p}$.

The Second Tangram Theorem

$A <:_n B$ if and only if $A <:^+ B$ and $B <:^- A$

The Second Blame Corollary

Let M be a term where $N : A \xrightarrow{p} B$ is the only subterm with label p .

If $A <:_n B$ then $M \not\rightarrow \text{blame } p$.

Let M be a term where $N : A \xrightarrow{p} B$ is the only subterm with label p .

If $B <:_n A$ then $M \not\rightarrow \text{blame } \bar{p}$.

Part VI

Blame and coercions

Lambda calculus with coercions

$$\frac{\Gamma \vdash M : A \quad c : A \Rightarrow B}{\Gamma \vdash M : A \overset{c}{\Rightarrow} B}$$

Coercion typing

$$\overline{\text{id}(A) : A \Rightarrow A}$$

$$\overline{(G!) : G \Rightarrow \star}$$

$$\overline{(p?G) : \star \Rightarrow G}$$

$$\frac{c : A' \Rightarrow A \quad d : B \Rightarrow B'}{(c \rightarrow d) : A \rightarrow B \Rightarrow A' \rightarrow B'}$$

$$\frac{c : A \Rightarrow B \quad d : B \Rightarrow C}{(c ; d) : A \Rightarrow C}$$

Blame safety is preserved and reflected

Let cast $A \xRightarrow{p} B$ translate to coercion c .

$A <:^+ B$ if and only if p does not appear in c .

$A <:^- B$ if and only if \bar{p} does not appear in c .

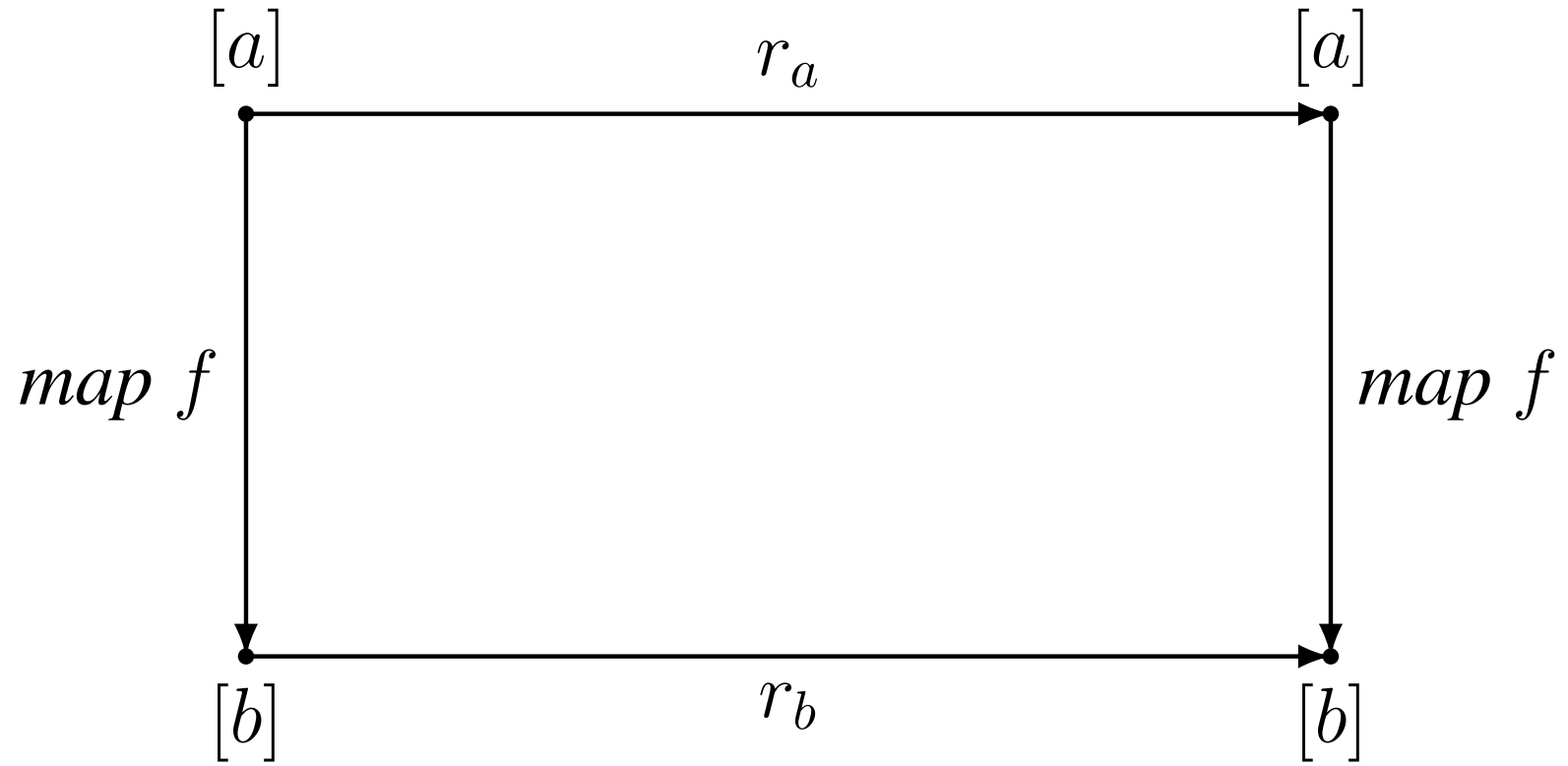
Part VII

Polymorphism

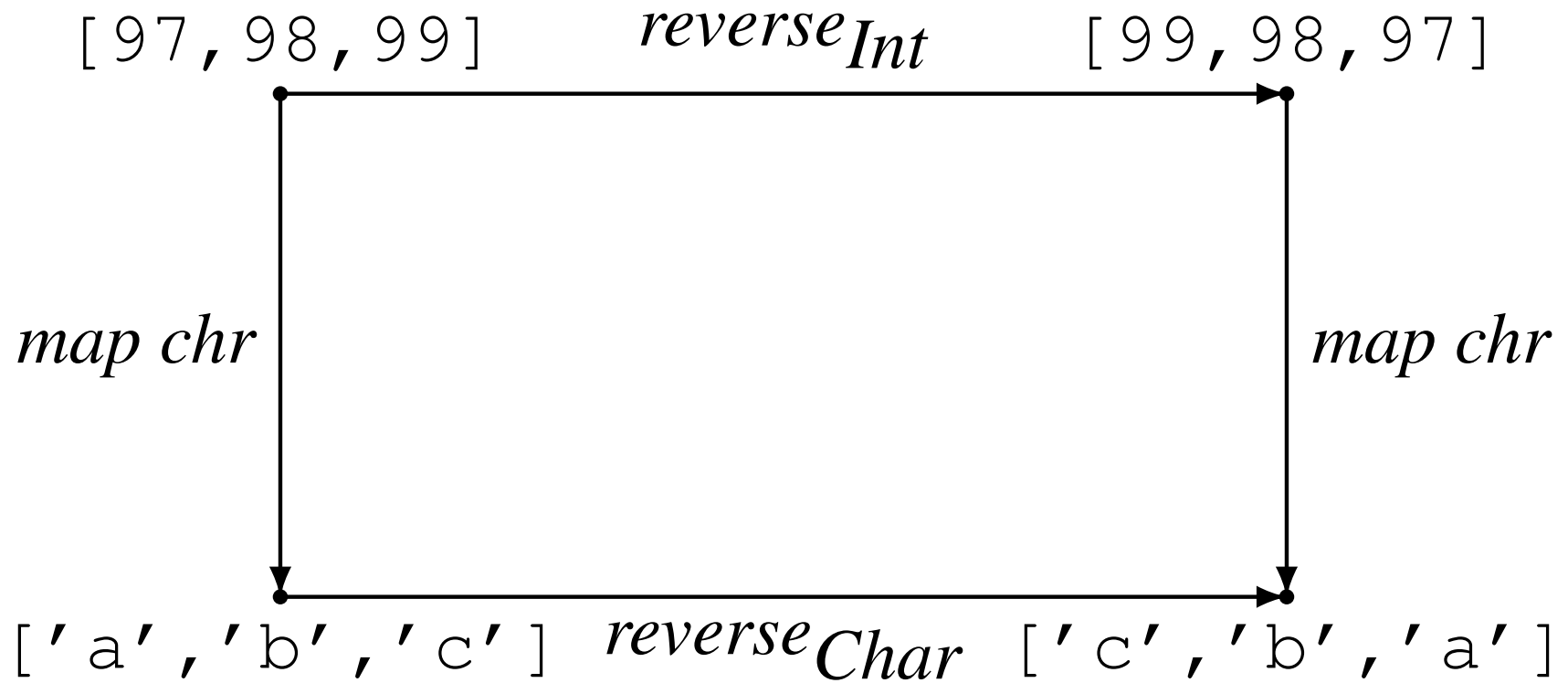
A magic trick

$$r :: [a] \longrightarrow [a]$$

Theorems for Free!



Theorems for Free!



Explicit binding

$$\frac{\Gamma, X:=A \vdash N : B \quad X \notin \text{ftv}(B)}{\Gamma \vdash \nu X:=A. N : B}$$

$$\frac{\Gamma \vdash N : B \quad (X:=A) \in \Gamma}{\Gamma \vdash N : B[X:=A]}$$

$$\frac{\Gamma \vdash N : B[X:=A] \quad (X:=A) \in \Gamma}{\Gamma \vdash N : B}$$

$$(\lambda X. N) A \longrightarrow \nu X:=A. N$$

Global store vs. Local bindings

George Neis, Derek Dreyer, and Andreas Rossberg. Non-parametric parametricity. ICFP 2009, Edinburgh.

Compatibility and reductions

$$\frac{}{X \prec X} \quad \frac{A \prec B}{A \prec \forall X. B} \quad X \notin \text{ftv}(A) \quad \frac{A[X:=\star] \prec B}{\forall X. A \prec B}$$

$$V : A \xRightarrow{p} (\forall X. B) \longrightarrow \Lambda X. (V : A \xRightarrow{p} B) \quad \text{if } X \notin \text{ftv}(A)$$

$$V : (\forall X. A) \xRightarrow{p} B \longrightarrow (V \star) : A[X:=\star] \xRightarrow{p} B$$

$$V : X \xRightarrow{p} \star \xRightarrow{q} X \longrightarrow V$$

$$V : X \xRightarrow{p} \star \xRightarrow{q} Y \longrightarrow \text{blame } q \quad \text{if } X \neq Y$$

Instantiate

$$V : \forall X. A \xRightarrow{p} B \longrightarrow V \star : A[X := \star] \xRightarrow{p} B$$

$$K = \lambda x : X. \lambda y : X. x$$

$$\begin{aligned} & (((\lambda X. K) : \forall X. X \rightarrow X \rightarrow X \xRightarrow{p} \star \rightarrow \star \rightarrow \star) \ 42 \ 7 \\ \longrightarrow & (((\lambda X. K) \star : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \star \rightarrow \star \rightarrow \star) \ 42 \ 7 \\ \longrightarrow & \nu X := \star. (K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \star \rightarrow \star \rightarrow \star) \ 42 \ 7 \\ \longrightarrow & \nu X := \star. (K (42 : \star \xRightarrow{\bar{p}} \star) (7 : \star \xRightarrow{\bar{p}} \star)) : \star \xRightarrow{p} \star \\ \longrightarrow & \nu X := \star. (42 : \star \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} \star \\ \longrightarrow & \nu X := \star. 42 \\ \longrightarrow & 42 \end{aligned}$$

★ is a Jack-of-all-Trades

$$V : \forall X. A \xRightarrow{p} B \longrightarrow V \star : A[X := \star] \xRightarrow{p} B$$

$$K = \lambda x : X. \lambda y : X. x$$

$$\begin{aligned} & ((\lambda X. K) : \forall X. X \rightarrow X \rightarrow X \xRightarrow{p} \text{Int} \rightarrow \text{Int} \rightarrow \text{Int}) \ 42 \ 7 \\ \longrightarrow & ((\lambda X. K) \star : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \text{Int} \rightarrow \text{Int} \rightarrow \text{Int}) \ 42 \ 7 \\ \longrightarrow & \nu X := \star. (K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \text{Int} \rightarrow \text{Int} \rightarrow \text{Int}) \ 42 \ 7 \\ \longrightarrow & \nu X := \star. (K (42 : \text{Int} \xRightarrow{\bar{p}} \star) (7 : \text{Int} \xRightarrow{\bar{p}} \star)) : \star \xRightarrow{p} \text{Int} \\ \longrightarrow & \nu X := \star. (42 : \text{Int} \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} \text{Int} \\ \longrightarrow & \nu X := \star. 42 \\ \longrightarrow & 42 \end{aligned}$$

... but master of none

$$V : \forall X. A \xRightarrow{p} B \longrightarrow V \star : A[X := \star] \xRightarrow{p} B$$

$$K = \lambda x : X. \lambda y : X. x$$

$$\begin{aligned} & ((\lambda X. K) : \forall X. X \rightarrow X \rightarrow X \xRightarrow{p} \text{Int} \rightarrow \text{Bool} \rightarrow \text{Int}) \ 42 \ \text{true} \\ \longrightarrow & ((\lambda X. K) \star : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \text{Int} \rightarrow \text{Bool} \rightarrow \text{Int}) \ 42 \ \text{true} \\ \longrightarrow & \nu X := \star. (K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \text{Int} \rightarrow \text{Bool} \rightarrow \text{Int}) \ 42 \ \text{true} \\ \longrightarrow & \nu X := \star. (K (42 : \text{Int} \xRightarrow{\bar{p}} \star) (\text{true} : \text{Bool} \xRightarrow{\bar{p}} \star)) : \star \xRightarrow{p} \text{Int} \\ \longrightarrow & \nu X := \star. (42 : \text{Int} \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} \text{Int} \\ \longrightarrow & \nu X := \star. 42 \\ \longrightarrow & 42 \end{aligned}$$

Generalise

$$V : A \xRightarrow{p} \forall X. B \longrightarrow \Lambda X. (V : A \xRightarrow{p} B) \quad X \text{ not free in } A$$

$$K = \lambda x : \star. \lambda y : \star. x$$

$$(K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \forall X. \forall Y. X \rightarrow Y \rightarrow X) \text{ Int Int 42 7}$$

$$\longrightarrow (\Lambda X. \Lambda Y. (K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} X \rightarrow Y \rightarrow X)) \text{ Int Int 42 7}$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (K : \star \rightarrow \star \rightarrow \star \xRightarrow{p} X \rightarrow Y \rightarrow X) \text{ 42 7}$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (K (42 : X \xRightarrow{\bar{p}} \star) (7 : Y \xRightarrow{\bar{p}} \star)) : \star \xRightarrow{p} X$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (42 : X \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} X$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. 42$$

$$\longrightarrow 42$$

Enforcing semantic parametricity

$$V : A \xRightarrow{p} \forall X. B \longrightarrow \Lambda X. (V : A \xRightarrow{p} B) \quad X \text{ not free in } A$$

$$K' = \lambda x : \star. \lambda y : \star. y$$

$$(K' : \star \rightarrow \star \rightarrow \star \xRightarrow{p} \forall X. \forall Y. X \rightarrow Y \rightarrow X) \text{ Int Int 42 7}$$

$$\longrightarrow (\Lambda X. \Lambda Y. (K' : \star \rightarrow \star \rightarrow \star \xRightarrow{p} X \rightarrow Y \rightarrow X)) \text{ Int Int 42 7}$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (K' : \star \rightarrow \star \rightarrow \star \xRightarrow{p} X \rightarrow Y \rightarrow X) \text{ 42 7}$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (K' (42 : X \xRightarrow{\bar{p}} \star) (7 : Y \xRightarrow{\bar{p}} \star)) : \star \xRightarrow{p} X$$

$$\longrightarrow \nu X := \text{Int}. \nu Y := \text{Int}. (7 : Y \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} X$$

$$\longrightarrow \text{blame } p$$

You can't look inside a seal

$$(V : G \xRightarrow{p} \star) \text{ is } H \longrightarrow \begin{cases} \text{true} & \text{if } G = H \\ \text{false} & \text{if } G \neq H \text{ and } G \neq X \\ \text{blame } p_{\text{is}} & \text{if } G = X \end{cases}$$

$\text{test} = \lambda x. \text{if } (x \text{ is Int}) \text{ then } [x + 1] \text{ else } x$

$(\text{test} : \star \rightarrow \star \xRightarrow{p} \forall X. X \rightarrow X) \text{ Int } 2$

$\longrightarrow \nu X := \text{Int}. \text{test } (2 : X \xRightarrow{\bar{p}} \star) : \star \xRightarrow{p} X$

$\longrightarrow \nu X := \text{Int}. \text{if } ((2 : X \Rightarrow \star) \text{ is Int}) \text{ then } \dots \text{ else } \dots$

$\longrightarrow \text{blame } p_{\text{is}}$

Part VIII

Threesomes

Even and odd, untyped

```
[let
  even* =  $\lambda n.$  if  $n = 0$ 
    then true
    else odd* ( $n - 1$ )
  odd* =  $\lambda n.$  if  $n = 0$ 
    then false
    else even* ( $n - 1$ )
in
  even* 2]
```

Even and odd, partly typed

let

$even^* = [\lambda n. \text{if } n = 0$
 then true
 else $odd^* (n - 1)]$

$odd = \lambda n : \text{Int}. \text{if } n = 0$
 then false
 else $even (n - 1)$

$even = even^* : \star \xRightarrow{p} \text{Int} \rightarrow \text{Int}$

$odd^* = odd : \text{Int} \rightarrow \text{Int} \xRightarrow{q} \star$

in

$[even^* 2]$

Goodbye tail recursion!

$[even^* 4]$

→

$(odd\ 3) : Bool \xRightarrow{q} *$

→

$[even^* 2] : * \xRightarrow{p} Bool \xRightarrow{q} *$

→

$(odd\ 1) : Bool \xRightarrow{q} * \xRightarrow{p} Bool \xRightarrow{q} *$

→

$[even^* 0] : * \xRightarrow{p} Bool \xRightarrow{q} * \xRightarrow{p} Bool \xRightarrow{q} *$

→

$[true]$

Recovering tail recursion!

$[even^* 4]$

→

$(odd\ 3) : Bool \xRightarrow{q} *$

→

$[even^* 2] : * \xRightarrow{p} Bool \xRightarrow{q} *$

→

$(odd\ 1) : Bool \xRightarrow{q} *$

→

$[even^* 0] : * \xRightarrow{p} Bool \xRightarrow{q} *$

→

$[true]$

Part IX

Twosomes: a small change to blame calculus

Add the empty type

Types $A, B, C ::= \iota \mid A \rightarrow B \mid \star \mid \perp$

Every type is incompatible with \perp .

$$A \not\sim \perp \quad \perp \not\sim A$$

There are no values of type \perp .

Allow casts between any types

Change one type rule

$$\frac{\Gamma \vdash M : A \quad A \prec B}{\Gamma \vdash (M : A \xRightarrow{p} B) : B}$$

becomes

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash (M : A \xRightarrow{p} B) : B}$$

Add one reduction

$$V : A \xRightarrow{p} \perp \longrightarrow \text{blame } p$$

Naive subtype

$$\overline{A <{:}_n \star}$$

$$\overline{l <{:}_n l}$$

$$\frac{A <{:}_n A' \quad B <{:}_n B'}{A \rightarrow B <{:}_n A' \rightarrow B'}$$

$$\overline{\perp <{:}_n A}$$

Meet

$$A \& \star = A$$

$$\star \& A = A$$

$$\iota \& \iota = \iota$$

$$(A \rightarrow B) \& (A' \rightarrow B') = (A \& A') \rightarrow (B \& B')$$

$$A \& B = \perp \quad \text{otherwise}$$

$A \& B$ is the greatest lower bound of A and B .

$C <:_n A \& B$ if and only if $C <:_n A$ and $C <:_n B$

First factoring theorem

$$V : A \xRightarrow{p} B = V : A \xRightarrow{p} A \& B \xRightarrow{p} B$$

An example

$$\begin{aligned} V : (\text{Int} \rightarrow \star) &\xRightarrow{p} (\star \rightarrow \text{Bool}) \\ &= \\ V : (\text{Int} \rightarrow \star) &\xRightarrow{p} (\text{Int} \rightarrow \text{Bool}) \xRightarrow{p} (\star \rightarrow \text{Bool}) \end{aligned}$$

Second factoring theorem

$$V : A \xRightarrow{p} B \quad = \quad V : A \xRightarrow{p} \star \xRightarrow{p} B$$

An example

$$\begin{aligned} V : (\text{Int} \rightarrow \star) &\xRightarrow{p} (\star \rightarrow \text{Bool}) \\ &= \\ V : (\text{Int} \rightarrow \star) &\xRightarrow{p} \star \xRightarrow{p} (\star \rightarrow \text{Bool}) \end{aligned}$$

Third factoring theorem

$$V : A \xRightarrow{p} B \quad = \quad V : A \xRightarrow{p} C \xRightarrow{p} B$$

whenever $A \& B <{:}_n C$

This result subsumes the other two.

Take $C = A \& B$ for the first theorem.

Take $C = \star$ for the second theorem.

Part X

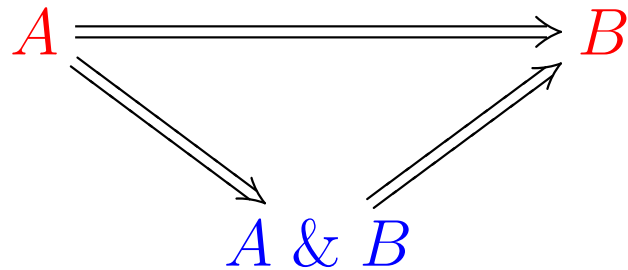
Threesomes, without blame

Threesomes

Let A, B, C, P, Q range over types.

$$V : A \xRightarrow{P} B \quad =_{\text{def}} \quad V : A \Rightarrow P \Rightarrow B$$

whenever $P <:_n A$ and $P <:_n B$.

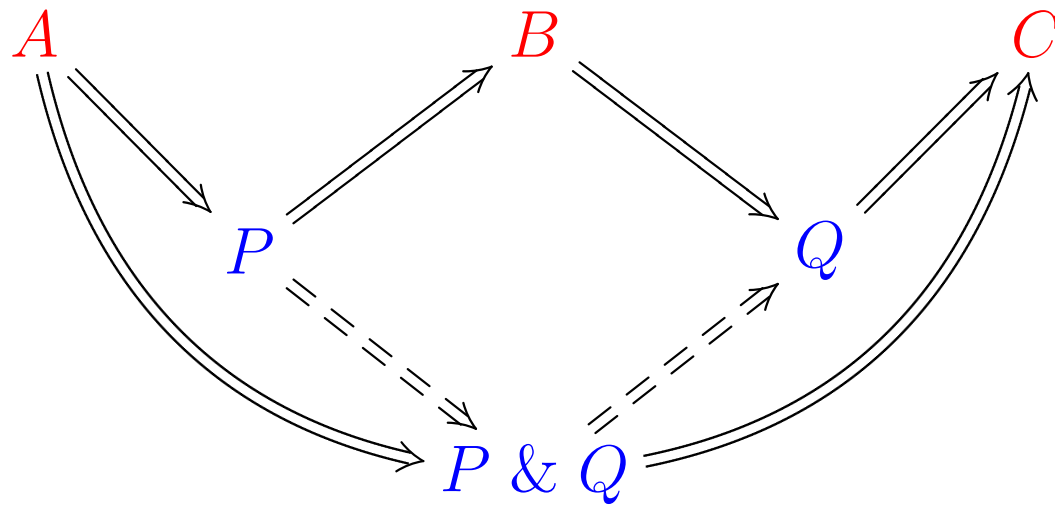


Combining threesomes

$$\begin{aligned} & V : A \xRightarrow{P} B \xRightarrow{Q} C \\ = & \quad \text{Def'n: } P <:_n A, P <:_n B, Q <:_n B, Q <:_n C \\ & V : A \Rightarrow P \Rightarrow B \Rightarrow Q \Rightarrow C \\ = & \quad \text{3'rd thm, } P \& Q <:_n B \\ & V : A \Rightarrow P \Rightarrow Q \Rightarrow C \\ = & \quad \text{1'st thm} \\ & V : A \Rightarrow P \Rightarrow P \& Q \Rightarrow Q \Rightarrow C \\ = & \quad \text{3'rd thm, } A \& P \& Q <:_n P, P \& Q \& C <:_n Q \\ & V : A \Rightarrow P \& Q \Rightarrow C \\ = & \quad \text{Def'n: } P \& Q <:_n A, P \& Q <:_n C \\ & V : A \xRightarrow{P\&Q} C \end{aligned}$$

Combining threesomes

$$V : A \xrightarrow{P} B \xrightarrow{Q} C \longrightarrow V : A \xrightarrow{P \& Q} C$$



Recovering tail recursion!

$[even^* 4]$

→

$(odd\ 3) : Bool \xRightarrow{Bool} *$

→

$[even^* 2] : * \xRightarrow{Bool} *$

→

$(odd\ 1) : Bool \xRightarrow{Bool} *$

→

$[even^* 0] : * \xRightarrow{Bool} *$

→

$[true]$

Part XI

Coercions and Threesomes

Coercions and threesomes

Coercions $c, d ::= \text{id}(A) \mid G! \mid p?G \mid c \rightarrow d \mid c; d$

Threesome coercions $s, t ::= \star \mid p?i \mid i$

Injection coercions $i ::= g! \mid g \mid \perp^{GpH}$

Ground coercions $g, h ::= \iota \mid s \rightarrow t$

$c : A \Rightarrow B$

$s : A \Rightarrow B$

$i : A \Rightarrow B$ implies $A \neq \star$

$g : A \Rightarrow B$ implies $A \neq \star$ and $B \neq \star$

Threesomes are coercions

$$\text{Int} \rightarrow \star \quad (\bar{p} ? \text{Int}) \xrightarrow{\quad} (p ? \text{Bool}) \quad \star \rightarrow \text{Bool}$$

$$\star \rightarrow \text{Bool} \quad (\text{Int} !) \xrightarrow{\quad} (\text{Bool} !) \quad \text{Int} \rightarrow \star$$

$$\star \rightarrow \star \quad (\text{Int} !) \xrightarrow{\quad} (p ? \text{Bool}) \quad \text{Int} \rightarrow \text{Bool}$$

$$\text{Int} \rightarrow \text{Bool} \quad (\bar{p} ? \text{Int}) \xrightarrow{\quad} (\text{Bool} !) \quad \star \rightarrow \star$$

Threesomes are coercions

$$\text{Int} \rightarrow \text{Int} \quad \perp^{\text{Bool } \bar{p} \text{ Int}} \xrightarrow{\perp} \perp^{\text{Int } p \text{ Bool}} \quad \text{Bool} \rightarrow \text{Bool}$$

$$\text{Int} \rightarrow \star \quad \xrightarrow{\text{Int} \rightarrow \star} \quad \text{Int} \rightarrow \star$$

Threesome typing

$$\frac{}{\iota : \iota \Rightarrow \iota} \quad \frac{}{\star : \star \Rightarrow \star}$$

$$\frac{g : A \Rightarrow G}{(g!) : A \Rightarrow \star}$$

$$\frac{i : G \Rightarrow A}{(p?i) : \star \Rightarrow A}$$

$$\frac{s : A' \Rightarrow A \quad t : B \Rightarrow B'}{(s \rightarrow t) : A \rightarrow B \Rightarrow A' \rightarrow B'}$$

$$\frac{\star \neq A \quad A \prec G \quad G \neq H}{(\perp^{GpH}) : A \Rightarrow B}$$

Threesome composition

$$\iota \circledast \iota = \iota$$

$$(s \rightarrow t) \circledast (s' \rightarrow t') = (s' \circledast s) \rightarrow (t \circledast t')$$

$$\star \circledast t = t$$

$$(g!) \circledast \star = g!$$

$$(p?i) \circledast t = p?(i \circledast t)$$

$$g \circledast (h!) = (g \circledast h)!$$

$$(g!) \circledast (p?i) = \begin{cases} g \circledast i & \text{if } G = H \\ \perp^{GpH} & \text{if } G \neq H \end{cases} \quad \text{where } \begin{array}{l} g : A \Rightarrow G \\ i : H \Rightarrow B \end{array}$$

$$\perp^{GpH} \circledast s = \perp^{GpH}$$

$$g \circledast \perp^{GpH} = \perp^{GpH}$$

Part XII

Conclusion

Publications

- Wadler and Findler
Well-typed programs can't be blamed
ESOP 2009
- Siek and Wadler
Threesomes, with and without blame
POPL 2010
- Ahmed, Findler, Siek, and Wadler
Blame for all
POPL 2011
- Siek, Thiemann, Wadler
Blame, coercions, and threesomes, precisely
draft

A new slogan for type safety

Milner (1978):

Well-typed programs can't go wrong.

Felleisen and Wright (1994); Harper (2002):

Well-typed programs don't get stuck.

Wadler and Findler (2008):

Well-typed programs can't be blamed.