Well-typed programs can’t be blamed

Philip Wadler
University of Edinburgh

Summer School on Trends in Computing
Tarragona, 25–26 July 2013

Collaborators

Ahmal Ahmed

Northeastern University

Robert Bruce Findler

Northwestern University

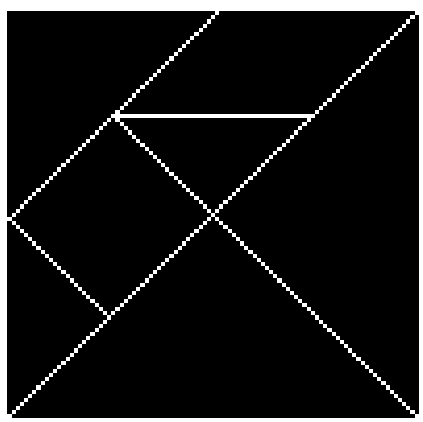
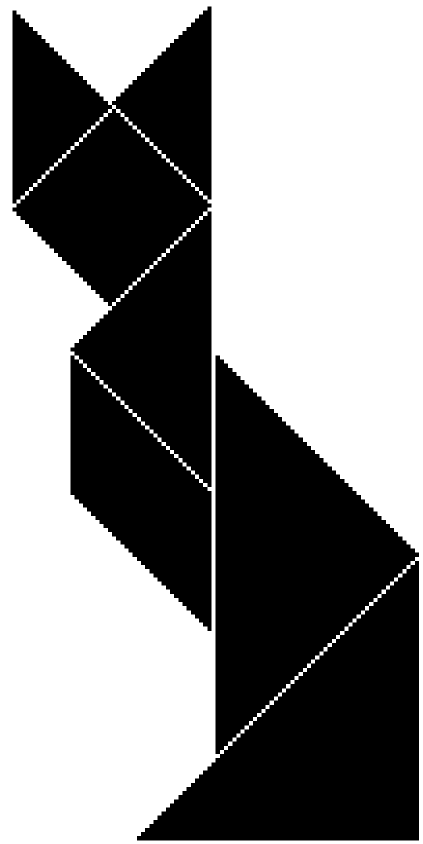
Jacob Matthews

Google

Jeremy Siek

University of Colorado at Boulder


Scotland
Housesteads
Wallsend
Chesters
Vindolanda
Birdoswald
Carlisle
England

A repeated theme

Thatte (1988):
Partial types

Henglein (1994):
Dynamic typing

Findler and Felleisen (2002):
Contracts

Flanagan (2006):
Hybrid types

Siek and Taha (2006):
Gradual types

A repeated theme

Dynamics in .Net (C#, Visual Basic)

Perl 6.0

Javascript

Dart

Part I

Evolving a program

An untyped program

$$
\begin{array}{l}
\ulcorner\texttt{let} \\
\quad x = \texttt{2} \\
\quad f = \lambda y.\, y + \texttt{1} \\
\quad h = \lambda g.\, g\,(g\,x) \\
\texttt{in} \\
\quad h\,f\urcorner
\end{array}
$$

$$\longrightarrow$$

$$\ulcorner\texttt{4}\urcorner$$

A typed program

$$
\begin{aligned}
&\texttt{let} \\
&\quad x = \texttt{2} \\
&\quad f = \lambda y : \texttt{Int}.\, y + \texttt{1} \\
&\quad h = \lambda g : \texttt{Int} \to \texttt{Int}.\, g\,(g\,x) \\
&\texttt{in} \\
&\quad h\,f
\end{aligned}
$$

$$\longrightarrow$$

$$\texttt{4} : \texttt{Int}$$

A partly typed program—narrowing

$$\begin{array}{l}
\texttt{let} \\
\quad x = \texttt{2} \\
\quad f = \lceil \lambda y.\, y + \texttt{1} \rceil : \star \stackrel{p}{\Rightarrow} \texttt{Int} \to \texttt{Int} \\
\quad h = \lambda g : \texttt{Int} \to \texttt{Int}.\, g\,(g\,x) \\
\texttt{in} \\
\quad h\,f
\end{array}$$

$$\longrightarrow$$

$$\texttt{4} : \texttt{Int}$$

A partly typed program—narrowing

$$
\begin{array}{l}
\texttt{let} \\
\quad x = \texttt{2} \\
\quad f = \lceil \lambda y.\, \texttt{false} \rceil : \star \stackrel{p}{\Longrightarrow} \texttt{Int} \to \texttt{Int} \\
\quad h = \lambda g : \texttt{Int} \to \texttt{Int}.\, g\,(g\,x) \\
\texttt{in} \\
\quad h\ f
\end{array}
$$

$$\longrightarrow$$

$$\texttt{blame}\ p$$

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

Another partly typed program—widening

$$
\begin{array}{l}
\texttt{let} \\
\quad x = \lceil \texttt{2} \rceil \\
\quad f = (\lambda y : \texttt{Int}.\, y + \texttt{1}) : \texttt{Int} \to \texttt{Int} \stackrel{p}{\Longrightarrow} \star \\
\quad h = \lceil \lambda g.\, g\,(g\,x) \rceil \\
\texttt{in} \\
\quad \lceil h\,f \rceil
\end{array}
$$

$$\longrightarrow$$

$$\lceil 4 \rceil$$

Another partly typed program—widening

$$
\begin{array}{l}
\texttt{let} \\
\quad x = \lceil \texttt{true} \rceil \\
\quad f = (\lambda y : \texttt{Int}.\, y + 1) : \texttt{Int} \to \texttt{Int} \stackrel{p}{\Rightarrow} \star \\
\quad h = \lceil \lambda g.\, g\,(g\,x) \rceil \\
\texttt{in} \\
\quad \lceil h\ f \rceil
\end{array}
$$

$$\longrightarrow$$

$$\texttt{blame}\ \bar{p}$$

Negative (contravariant): blame the context containg the cast

Part II

Untyped and supertyped

Untyped = Uni-typed

$$
\begin{array}{lcll}
\lceil x \rceil & = & x & \\
\lceil c \rceil & = & c : A \overset{p}{\Rightarrow} \star & \text{if } \mathbf{ty}(c) = A \\
\lceil op(\vec{M}) \rceil & = & op(\lceil \vec{M} \rceil : \vec{\star} \overset{\vec{p}}{\Rightarrow} \vec{A}) : B \overset{p}{\Rightarrow} \star & \text{if } \mathbf{ty}(op) = \vec{A} \to B \\
\lceil \lambda x.\, N \rceil & = & (\lambda x : \star.\, \lceil N \rceil) : \star \to \star \Rightarrow \star & \\
\lceil L\ M \rceil & = & (\lceil L \rceil : \star \overset{p}{\Rightarrow} \star \to \star)\ \lceil M \rceil &
\end{array}
$$

(slogan due to Bob Harper)

Contracts

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

$$\texttt{let}$$

$$x = \texttt{2} : \texttt{Int} \overset{p}{\Longrightarrow} \texttt{Nat}$$

$$f = (\lambda y : \texttt{Int}.\, y + \texttt{1}) : \texttt{Int} \rightarrow \texttt{Int} \overset{q}{\Longrightarrow} \texttt{Nat} \rightarrow \texttt{Nat}$$

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

$$\texttt{in}$$

$$h\ f$$

$$\longrightarrow$$

$$\texttt{4} : \texttt{Nat}$$

Part III

The Blame Game

Reductions (simplified)

$$
\begin{array}{rcl}
v : \iota \Rightarrow \star \overset{p}{\Rightarrow} \iota & \longrightarrow & v \\
v : \iota \Rightarrow \star \overset{p}{\Rightarrow} \iota' & \longrightarrow & \texttt{blame}\ p \quad \textbf{if}\ \iota \neq \iota' \\
(v : A \to B \overset{p}{\Rightarrow} A' \to B')\ w & \longrightarrow & (v\ (w : A' \overset{\bar{p}}{\Rightarrow} A) : B \overset{p}{\Rightarrow} B')
\end{array}
$$

Blame

$$\lceil \mathtt{2} \rceil : \star \overset{p}{\Longrightarrow} \mathtt{Int}$$

$$=$$

$$\mathtt{2} : \mathtt{Int} \Rightarrow \star \overset{p}{\Longrightarrow} \mathtt{Int}$$

$$\longrightarrow$$

$$\mathtt{2}$$

$$\lceil \mathtt{true} \rceil : \star \overset{p}{\Longrightarrow} \mathtt{Int}$$

$$=$$

$$\mathtt{true} : \mathtt{Bool} \Rightarrow \star \overset{p}{\Longrightarrow} \mathtt{Int}$$

$$\longrightarrow$$

$$\mathtt{blame}\ p$$

The Blame Game—widening

$$((\lambda y : \texttt{Int}.\, y + 1) : \texttt{Int} \to \texttt{Int} \overset{p}{\Longrightarrow} \star \to \star)\; \lceil \texttt{2} \rceil$$

$$\longrightarrow$$

$$(\lambda y : \texttt{Int}.\, y + 1)\; (\lceil \texttt{2} \rceil : \star \overset{\bar{p}}{\Longrightarrow} \texttt{Int}) : \texttt{Int} \overset{p}{\Longrightarrow} \star$$

$$\longrightarrow$$

$$\lceil \texttt{3} \rceil$$

The Blame Game—widening

$$((\lambda y : \texttt{Int}.\, y + 1) : \texttt{Int} \to \texttt{Int} \stackrel{p}{\Rightarrow} \star \to \star)\ \lceil \texttt{true} \rceil$$

$$\longrightarrow$$

$$(\lambda y : \texttt{Int}.\, y + 1)\ (\lceil \texttt{true} \rceil : \star \stackrel{\overline{p}}{\Rightarrow} \texttt{Int}) : \texttt{Int} \stackrel{p}{\Rightarrow} \star$$

$$\longrightarrow$$

$$\texttt{blame}\ \overline{p}$$

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

The Blame Game—narrowing

$$((\lambda y : \star.\, \lceil y + \mathtt{1} \rceil) : \star \to \star \stackrel{p}{\Longrightarrow} \mathtt{Int} \to \mathtt{Int})\ \mathtt{2}$$

$$\longrightarrow$$

$$(\lambda y : \star.\, \lceil y + \mathtt{1} \rceil)\ (\mathtt{2} : \mathtt{Int} \stackrel{\bar{p}}{\Longrightarrow} \star) : \star \stackrel{p}{\Longrightarrow} \mathtt{Int}$$

$$\longrightarrow$$

$$\mathtt{3}$$

The Blame Game—narrowing

$$((\lambda y : \star.\, \lceil \texttt{false} \rceil) : \star \to \star \stackrel{p}{\Longrightarrow} \texttt{Int} \to \texttt{Int})\; \texttt{2}$$

$$\longrightarrow$$

$$(\lambda y : \star.\, \lceil \texttt{false} \rceil)\; (\texttt{2} : \texttt{Int} \stackrel{\bar{p}}{\Longrightarrow} \star) : \star \stackrel{p}{\Longrightarrow} \texttt{Int}$$

$$\longrightarrow$$

$$\texttt{blame}\; p$$

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

Part IV

Blame calculus in detail

Notation

It took us four years to find the right notation!

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

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

$$s : A \stackrel{p}{\Rightarrow} B$$

We want composition to be easy to read:

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

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

$$s : A \stackrel{p}{\Rightarrow} B : B \stackrel{q}{\Rightarrow} C$$

And there is a convenient abbreviation:

$$s : A \stackrel{p}{\Rightarrow} B \stackrel{q}{\Rightarrow} C$$

Syntax

Blame labels	p, q		
Base types	ι		
Types	A, B, C	$::=$	$\iota \mid A \to B \mid \star$
Ground types	G, H	$::=$	$\iota \mid \star \to \star$
Terms	s, t	$::=$	$c \mid op(\vec{t}) \mid x \mid \lambda x{:}A.\, t \mid t\ s \mid$
			$s : A \stackrel{p}{\Rightarrow} B \mid s : G \Rightarrow \star \mid \texttt{blame}\ p$
Environments	Γ	$::=$	$\cdot \mid \Gamma, x : A$
Values	v, w	$::=$	$c \mid \lambda x{:}A.\, t \mid v : G \Rightarrow \star$
Contexts	E	$::=$	$[\cdot] \mid op(\vec{v}, E, \vec{t}) \mid E\ s \mid v\ E \mid$
			$E : A \stackrel{p}{\Rightarrow} B \mid E : G \Rightarrow \star$

Blame calculus: Compatibility

$$A \prec \star \qquad \star \prec B$$

$$\iota \prec \iota$$

$$\frac{A' \prec A \qquad B \prec B'}{A \to B \prec A' \to B'}$$

Types

$$\frac{\mathrm{ty}(c) = \iota}{\Gamma \vdash c : \iota} \qquad \frac{\Gamma \vdash \vec{t} : \vec{A} \qquad \mathrm{ty}(op) = \vec{A} \to B}{\Gamma \vdash op(\vec{t}) : B} \qquad \frac{x : A \in \Gamma}{\Gamma \vdash x : A}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x{:}A.\, t : A \to B} \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash s : A}{\Gamma \vdash t\, s : B}$$

$$\frac{\Gamma \vdash s : A \qquad A \prec B}{\Gamma \vdash (s : A \stackrel{p}{\Rightarrow} B) : B} \qquad \frac{\Gamma \vdash s : G}{\Gamma \vdash (s : G \Rightarrow \star) : \star}$$

$$\Gamma \vdash \texttt{blame}\, p : A$$

Beta, Delta, contextual closure

$$
\begin{array}{rcl}
(\lambda x{:}A.\, t)\, v & \longrightarrow & t[x{:=}v] \\
\boldsymbol{op}(\vec{v}\,) & \longrightarrow & \delta(\boldsymbol{op}, \vec{v}\,)
\end{array}
$$

$$
\frac{s \longrightarrow t}{E[s] \longrightarrow E[t]} \qquad \frac{E \neq [\cdot]}{E[\texttt{blame}\ p] \longrightarrow \texttt{blame}\ p}
$$

Wrap, Id, Ground, Collapse, Conflict

$$
\begin{array}{rcll}
v : A \to B \stackrel{p}{\Rightarrow} A' \to B' & \longrightarrow & \lambda x' {:} A'.\, (v\ (x' : A' \stackrel{\overline{p}}{\Rightarrow} A) : B \stackrel{p}{\Rightarrow} B') & \\
v : \iota \stackrel{p}{\Rightarrow} \iota & \longrightarrow & v & \\
v : A \stackrel{p}{\Rightarrow} \star & \longrightarrow & v : A \stackrel{p}{\Rightarrow} G \Rightarrow \star & \text{if } A \prec G \text{ and } A \neq \star \\
v : G \Rightarrow \star \stackrel{p}{\Rightarrow} A & \longrightarrow & v : G \stackrel{p}{\Rightarrow} A & \text{if } G \prec A \\
v : G \Rightarrow \star \stackrel{p}{\Rightarrow} A & \longrightarrow & \mathtt{blame}\ p & \text{if } G \not\prec A
\end{array}
$$

Part V

Subtyping

$$<: \qquad <:^{+} \qquad <:^{-} \qquad <:_{n}$$

Subtype

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

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

$$\frac{A' <: A \qquad B <: B'}{A \to B <: A' \to B'}$$

Example:

$$\frac{\dfrac{\dfrac{}{\texttt{Int} <: \texttt{Int}}}{\texttt{Int} <: \star} \qquad \dfrac{\dfrac{}{\texttt{Int} <: \texttt{Int}}}{\texttt{Int} <: \star}}{\star \to \texttt{Int} <: \texttt{Int} \to \star}$$

Positive subtype—widening

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

$$\frac{}{\iota <: \iota}$$

$$\frac{A' <:^{-} A \qquad B <:^{+} B'}{A \to B <:^{+} A' \to B'}$$

Example:

$$\frac{\dfrac{}{\star <:^{-} \mathtt{Int}} \qquad \dfrac{}{\mathtt{Int} <:^{+} \star}}{\mathtt{Int} \to \mathtt{Int} <: \star \to \star}$$

Negative subtype—narrowing

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

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

$$\frac{A' <:^{+} A \qquad B <:^{-} B'}{A \to B <:^{-} A' \to B'}$$

Example:

$$\frac{\dfrac{}{\texttt{Int} <:^{+} \star} \qquad \dfrac{}{\star <:^{-} \texttt{Int}}}{\star \to \star <:^{-} \texttt{Int} \to \texttt{Int}}$$

Naive subtype

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

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

$$\frac{A <:_n A' \qquad B <:_n B'}{A \to B <:_n A' \to B'}$$

Example:

$$\frac{\dfrac{}{\texttt{Int} <:_n \star} \qquad \dfrac{}{\texttt{Int} <:_n \star}}{\texttt{Int} \to \texttt{Int} <: \star \to \star}$$

Part VI

The Blame Theorem

Safety

$$\frac{}{x \ \mathrm{sf}\ p} \qquad \frac{t \ \mathrm{sf}\ p}{\lambda x.\, t \ \mathrm{sf}\ p} \qquad \frac{s \ \mathrm{sf}\ p \qquad t \ \mathrm{sf}\ p}{s\ t \ \mathrm{sf}\ p}$$

$$\frac{s \ \mathrm{sf}\ p \qquad A <:^{+} B}{s : A \stackrel{p}{\Rightarrow} B \ \mathrm{sf}\ p}$$

$$\frac{s \ \mathrm{sf}\ p \qquad A <:^{-} B}{s : A \stackrel{\bar{p}}{\Rightarrow} B \ \mathrm{sf}\ p}$$

$$\frac{s \ \mathrm{sf}\ p \qquad p \neq q \qquad \bar{p} \neq q}{s : A \stackrel{q}{\Rightarrow} B \ \mathrm{sf}\ p}$$

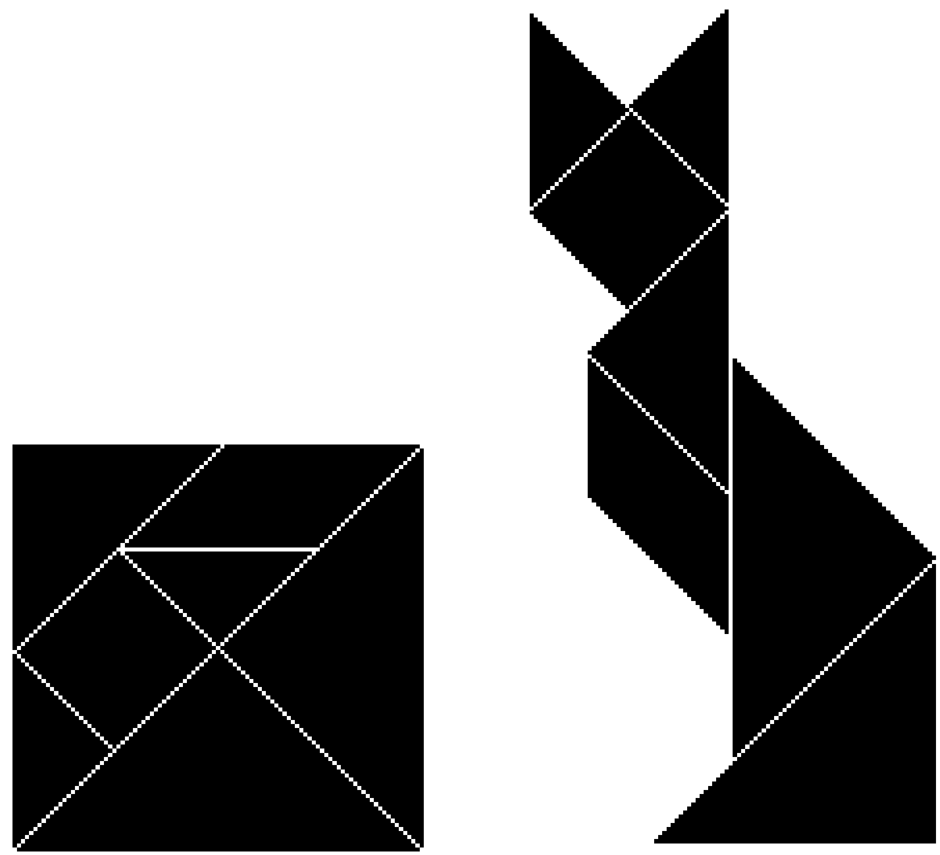
The Blame Theorem

Preservation

If s sf p and $s \longrightarrow t$ then t sf p.

Progress

If s sf p then $s \not\longrightarrow \texttt{blame}\ p$.



The First Tangram Theorem

$$A <: B \text{ if and only if } A <:^{+} B \text{ and } A <:^{-} B$$

The First Blame Corollary

Let t be a term where $s : A \stackrel{p}{\Longrightarrow} B$ is the only subterm with label p. If $A <: B$ then $t \not\longrightarrow \texttt{blame}\ p$ and $t \not\longrightarrow \texttt{blame}\ \bar{p}$.

The Second Tangram Theorem

$$A <:_n B \text{ if and only if } A <:^+ B \text{ and } B <:^- A$$

The Second Blame Corollary

Let t be a term where $s : A \stackrel{p}{\Rightarrow} B$ is the only subterm with label p. If $A <:_n B$ then $t \not\longrightarrow \texttt{blame}\ p$.

Let t be a term where $s : A \stackrel{p}{\Rightarrow} B$ is the only subterm with label p. If $B <:_n A$ then $t \not\longrightarrow \texttt{blame}\ \overline{p}$.

Part VII

Adding polymorphism

Explicit binding

$$\frac{\Gamma, X{:=}A \vdash t : B \qquad X \notin \mathrm{ftv}(B)}{\Gamma \vdash \nu X{:=}A.\, t : B}$$

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

$$(\Lambda X.\, v)\; A \quad\longrightarrow\quad \nu X{:=}A.\, v$$

Global store vs. Local bindings

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

Compatibility and reductions

$$X \prec X \qquad \frac{A \prec B}{A \prec \forall X.\, B}\; X \notin \mathrm{ftv}(A) \qquad \frac{A[X{:=}\star] \prec B}{\forall X.\, A \prec B}$$

$$\begin{array}{rcll}
v : A \stackrel{p}{\Rightarrow} (\forall X.\, B) & \longrightarrow & \Lambda X.\, v : A \stackrel{p}{\Rightarrow} B & \text{if } X \notin \mathrm{ftv} A \\
v : (\forall X.\, A) \stackrel{p}{\Rightarrow} B & \longrightarrow & v\,\star : A[X{:=}\star] \stackrel{p}{\Rightarrow} B & \\
v : X \stackrel{p}{\Rightarrow} \star \stackrel{q}{\Rightarrow} X & \longrightarrow & v & \\
v : X \stackrel{p}{\Rightarrow} \star \stackrel{q}{\Rightarrow} Y & \longrightarrow & \mathtt{blame}\; q & \text{if } X \neq Y
\end{array}$$

Instantiate

$$v : \forall X.\, A \overset{p}{\Rightarrow} B \;\longrightarrow\; v \star : A[X{:=}\star] \overset{p}{\Rightarrow} B$$

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

$$
\begin{array}{ll}
 & (((\Lambda X.\, K) : \forall X.\, X{\to}X{\to}X \overset{p}{\Rightarrow} \star{\to}\star{\to}\star)\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow & (((\Lambda X.\, K)\, \star : \star{\to}\star{\to}\star \overset{p}{\Rightarrow} \star{\to}\star{\to}\star)\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow & \nu X{:=}\star.\, (K : \star{\to}\star{\to}\star \overset{p}{\Rightarrow} \star{\to}\star{\to}\star)\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow & \nu X{:=}\star.\, (K\, (\mathtt{42} : \star \overset{\bar{p}}{\Rightarrow} \star)\, (\mathtt{7} : \star \overset{\bar{p}}{\Rightarrow} \star)) : \star \overset{p}{\Rightarrow} \star \\
\longrightarrow & \nu X{:=}\star.\, (\mathtt{42} : \star \overset{\bar{p}}{\Rightarrow} \star) : \star \overset{p}{\Rightarrow} \star \\
\longrightarrow & \nu X{:=}\star.\, \mathtt{42} \\
\longrightarrow & \mathtt{42}
\end{array}
$$

$\star$ is a Jack-of-all-Trades

$$v : \forall X.\, A \stackrel{p}{\Rightarrow} B \;\longrightarrow\; v\,\star : A[X{:=}\star] \stackrel{p}{\Rightarrow} B$$

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

$$\begin{aligned}
& ((\Lambda X.\, K) : \forall X.\, X{\to}X{\to}X \stackrel{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Int}{\to}\mathtt{Int})\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & ((\Lambda X.\, K)\,\star : \star{\to}\star{\to}\star \stackrel{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Int}{\to}\mathtt{Int})\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & \nu X{:=}\star.\, (K : \star{\to}\star{\to}\star \stackrel{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Int}{\to}\mathtt{Int})\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & \nu X{:=}\star.\, (K\ (\mathtt{42} : \mathtt{Int} \stackrel{\bar{p}}{\Rightarrow} \star)\ (\mathtt{7} : \mathtt{Int} \stackrel{\bar{p}}{\Rightarrow} \star)) : \star \stackrel{p}{\Rightarrow} \mathtt{Int} \\
\longrightarrow\ & \nu X{:=}\star.\, (\mathtt{42} : \mathtt{Int} \stackrel{\bar{p}}{\Rightarrow} \star) : \star \stackrel{p}{\Rightarrow} \mathtt{Int} \\
\longrightarrow\ & \nu X{:=}\star.\, \mathtt{42} \\
\longrightarrow\ & \mathtt{42}
\end{aligned}$$

… but master of none

$$v : \forall X.\, A \overset{p}{\Rightarrow} B \;\longrightarrow\; v\,\star : A[X{:=}\star] \overset{p}{\Rightarrow} B$$

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

$$
\begin{aligned}
& ((\Lambda X.\, K) : \forall X.\, X{\to}X{\to}X \overset{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Bool}{\to}\mathtt{Int})\; \mathtt{42}\; \mathtt{true} \\
\longrightarrow\; & ((\Lambda X.\, K)\, \star : \star{\to}\star{\to}\star \overset{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Bool}{\to}\mathtt{Int})\; \mathtt{42}\; \mathtt{true} \\
\longrightarrow\; & \nu X{:=}\star.\, (K : \star{\to}\star{\to}\star \overset{p}{\Rightarrow} \mathtt{Int}{\to}\mathtt{Bool}{\to}\mathtt{Int})\; \mathtt{42}\; \mathtt{true} \\
\longrightarrow\; & \nu X{:=}\star.\, (K\, (\mathtt{42} : \mathtt{Int} \overset{\bar{p}}{\Rightarrow} \star)\, (\mathtt{true} : \mathtt{Bool} \overset{\bar{p}}{\Rightarrow} \star)) : \star \overset{p}{\Rightarrow} \mathtt{Int} \\
\longrightarrow\; & \nu X{:=}\star.\, (\mathtt{42} : \mathtt{Int} \overset{\bar{p}}{\Rightarrow} \star) : \star \overset{p}{\Rightarrow} \mathtt{Int} \\
\longrightarrow\; & \nu X{:=}\star.\, \mathtt{42} \\
\longrightarrow\; & \mathtt{42}
\end{aligned}
$$

Generalise

$$v : A \stackrel{p}{\Rightarrow} \forall X.\, B \;\longrightarrow\; \Lambda X.\, (v : A \stackrel{p}{\Rightarrow} B) \qquad X \text{ not free in } A$$

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

$$\begin{array}{l}
\quad (K : \star\!\to\star\to\!\star \stackrel{p}{\Rightarrow} \forall X.\forall Y.\, X\!\to\! Y\!\to\! X)\ \texttt{Int}\ \texttt{Int}\ \texttt{42}\ \texttt{7} \\
\longrightarrow (\Lambda X.\, \Lambda Y.\, (K : \star\!\to\star\to\!\star \stackrel{p}{\Rightarrow} X\!\to\! Y\!\to\! X))\ \texttt{Int}\ \texttt{Int}\ \texttt{42}\ \texttt{7} \\
\longrightarrow \nu X{:=}\texttt{Int}.\, \nu Y{:=}\texttt{Int}.\, (K : \star\!\to\star\to\!\star \stackrel{p}{\Rightarrow} X\!\to\! Y\!\to\! X)\ \texttt{42}\ \texttt{7} \\
\longrightarrow \nu X{:=}\texttt{Int}.\, \nu Y{:=}\texttt{Int}.\, (K\ (\texttt{42} : X \stackrel{\bar{p}}{\Rightarrow} \star)\ (\texttt{7} : Y \stackrel{\bar{p}}{\Rightarrow} \star)) : \star \stackrel{p}{\Rightarrow} X \\
\longrightarrow \nu X{:=}\texttt{Int}.\, \nu Y{:=}\texttt{Int}.\, (\texttt{42} : X \stackrel{\bar{p}}{\Rightarrow} \star) : \star \stackrel{p}{\Rightarrow} X \\
\longrightarrow \nu X{:=}\texttt{Int}.\, \nu Y{:=}\texttt{Int}.\, \texttt{42} \\
\longrightarrow \texttt{42}
\end{array}$$

Enforcing semantic parametricity

$$v : A \overset{p}{\Rightarrow} \forall X.\, B \longrightarrow \Lambda X.\, (v : A \overset{p}{\Rightarrow} B) \qquad X \text{ not free in } A$$

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

$$\begin{aligned}
& (K' : \star \to \star \to \star \overset{p}{\Rightarrow} \forall X.\, \forall Y.\, X \to Y \to X)\ \mathtt{Int}\ \mathtt{Int}\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & (\Lambda X.\, \Lambda Y.\, (K' : \star \to \star \to \star \overset{p}{\Rightarrow} X \to Y \to X))\ \mathtt{Int}\ \mathtt{Int}\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & \nu X{:=}\mathtt{Int}.\, \nu Y{:=}\mathtt{Int}.\, (K' : \star \to \star \to \star \overset{p}{\Rightarrow} X \to Y \to X)\ \mathtt{42}\ \mathtt{7} \\
\longrightarrow\ & \nu X{:=}\mathtt{Int}.\, \nu Y{:=}\mathtt{Int}.\, (K'\ (\mathtt{42} : X \overset{\bar{p}}{\Rightarrow} \star)\ (\mathtt{7} : Y \overset{\bar{p}}{\Rightarrow} \star)) : \star \overset{p}{\Rightarrow} X \\
\longrightarrow\ & \nu X{:=}\mathtt{Int}.\, \nu Y{:=}\mathtt{Int}.\, (\mathtt{7} : Y \overset{\bar{p}}{\Rightarrow} \star) : \star \overset{p}{\Rightarrow} X \\
\longrightarrow\ & \mathtt{blame}\ p
\end{aligned}$$

Part VIII

Conclusion

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.