# Dynamic Types

CS 152 (Spring 2020)

Harvard University

Thursday, April 23, 2020

# Today, we will learn about

- Error propagation
- Exception handling
- Contracts

# Error-propagation semantics

# Error-propagation semantics (syntax)

$$\begin{aligned} e &::= x \mid \lambda x.\, e \mid e_1\ e_2 \mid n \mid e_1 + e_2 \mid \mathsf{Err} \\ v &::= n \mid \lambda x.\, e \mid \mathsf{Err} \end{aligned}$$

# Error-propagation semantics

$$E ::= E\ e \mid v\ E \mid E + e \mid v + E$$

$$\frac{e \longrightarrow e'}{E[e] \longrightarrow E[e']} \qquad \frac{}{(\lambda x.\, e)\ v \longrightarrow e\{v/x\}}\ v \neq \mathrm{Err}$$

$$\frac{}{n_1 + n_2 \longrightarrow n}\ n = n_1 + n_2$$

$$\frac{}{v_1\ v_2 \longrightarrow \mathrm{Err}}\ v_1 \neq \lambda x.\, e$$

$$\frac{}{v_1 + v_2 \longrightarrow \mathrm{Err}}\ v_1 \text{ or } v_2 \text{ not an integer}$$

# Error-propagation semantics (propagation)

$$\ldots \quad \frac{}{(\lambda x.\, e)\ \mathsf{Err} \longrightarrow \mathsf{Err}} \quad \ldots$$

# Error-propagation semantics (example)

$$\begin{aligned}
& 42 + ((\lambda f.\, \lambda n.\, f\ (n + 3))\ (\lambda x.\, x)\ (\lambda x.\, x)) \\
\longrightarrow & 42 + ((\lambda n.\, (\lambda x.\, x)\ (n + 3))\ (\lambda x.\, x)) \\
\longrightarrow & 42 + ((\lambda x.\, x)\ ((\lambda x.\, x) + 3)) \\
\longrightarrow & 42 + ((\lambda x.\, x)\ \mathsf{Err}) \\
\longrightarrow & 42 + \mathsf{Err} \\
\longrightarrow & \mathsf{Err}
\end{aligned}$$

# Dynamic Types

$$
\begin{aligned}
e &::= \cdots \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \\
&\mid \mathsf{is\_int?}\ e \mid \mathsf{is\_fun?}\ e \mid \mathsf{is\_bool?}\ e \\
v &::= \cdots \mid \mathsf{true} \mid \mathsf{false} \\
E &::= \cdots \mid \mathsf{if}\ E\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \\
&\mid \mathsf{is\_int?}\ E \mid \mathsf{is\_fun?}\ E \mid \mathsf{is\_bool?}\ E
\end{aligned}
$$

# Dynamic Types Semantics (. . .)

$$\frac{}{\textsf{if true then } e_2 \textsf{ else } e_3 \longrightarrow e_2}$$

$$\frac{}{\textsf{if false then } e_2 \textsf{ else } e_3 \longrightarrow e_3}$$

$$\frac{}{\textsf{if } v \textsf{ then } e_2 \textsf{ else } e_3 \longrightarrow \textsf{Err}}\ v \neq \textsf{true and } v \neq \textsf{false}$$

# Dynamic Types Semantics (. . .)

$$\frac{}{\textsf{is\_int? } n \longrightarrow \textsf{true}}$$

$$\frac{}{\textsf{is\_int? } v \longrightarrow \textsf{false}}\; v \textsf{ not an integer and } v \neq \textsf{Err}$$

# Dynamic Types Semantics (. . .)

$$\frac{}{\text{is\_fun? } \lambda x.\, e \longrightarrow \text{true}}$$

$$\frac{}{\text{is\_fun? } v \longrightarrow \text{false}}\ v \neq \lambda x.\, e \text{ and } v \neq \text{Err}$$

# Dynamic Types Semantics (. . .)

$$\frac{}{\textsf{is\_bool? } v \longrightarrow \textsf{true}}\; v = \textsf{true or } v = \textsf{false}$$

$$\frac{}{\textsf{is\_bool? } v \longrightarrow \textsf{false}}\; v \notin \{\textsf{true}, \textsf{false}, \textsf{Err}\}$$

# Dynamic Types Semantics (Error-Propagation Rules)

$$\frac{}{\text{is\_int? Err} \longrightarrow \text{Err}}$$

$$\frac{}{\text{is\_fun? Err} \longrightarrow \text{Err}}$$

$$\frac{}{\text{is\_bool? Err} \longrightarrow \text{Err}}$$

# Alternative Error-Propagation Rule

$$\frac{e \longrightarrow \mathsf{Err}}{E[e] \longrightarrow \mathsf{Err}}$$

# Exception handling

# Exception handling

$$
\begin{aligned}
e &::= \cdots \mid \mathsf{try}\ e_1\ \mathsf{catch}\ x.\ e_2 \mid \mathsf{raise}\ e \\
v &::= \cdots \mid \mathsf{Err}\ v \\
E &::= \cdots \mid \mathsf{try}\ E\ \mathsf{catch}\ x.\ e_2 \mid \mathsf{raise}\ E
\end{aligned}
$$

# Exception handling

$$\frac{}{\text{raise } v \longrightarrow \text{Err } v}\; v \neq \text{Err } v'$$

$$\frac{}{\text{raise } (\text{Err } v) \longrightarrow \text{Err } v}$$

$$\frac{}{\text{try Err } v \text{ catch } x.\ e_2 \longrightarrow e_2\{v/x\}}$$

$$\frac{}{\text{try } v \text{ catch } x.\ e_2 \longrightarrow v}\; v \neq \text{Err } v'$$

# Exception handling

- 0: a non-function value was applied
- 1: a non-integer value was used as an operand for addition
- 2: a non-boolean value was used as the test for a conditional

# Exception handling

$$\frac{}{v_1\ v_2 \longrightarrow \mathsf{Err}\ 0}\ v_1 \neq \lambda x.\, e \text{ and } v_1 \neq \mathsf{Err}\ v$$

$$\frac{}{(\mathsf{Err}\ v_1)\ v_2 \longrightarrow \mathsf{Err}\ v_1}$$

# Exception handling

$$\frac{}{v_1 + v_2 \longrightarrow \text{Err } 1} *$$

$*$ $v_1$ or $v_2$ not an integer and
$v_1 \neq \text{Err } v$ and $v_2 \neq \text{Err } v'$

$$\frac{}{(\text{Err } v_1) + v_2 \longrightarrow \text{Err } v_1}$$

$$\frac{}{n_1 + \text{Err } v \longrightarrow \text{Err } v}$$

# Exception handling

$$\frac{}{\text{if } v \text{ then } e_2 \text{ else } e_3 \longrightarrow \text{Err } 2}\ *$$

$*\ v \neq \text{true}$ and $v \neq \text{false}$ and $v \neq \text{Err } v'$

$$\frac{}{\text{if Err } v \text{ then } e_2 \text{ else } e_3 \longrightarrow \text{Err } v}$$

# Exception handling: Example program

$$\text{let } \mathit{foo} = \lambda x.\, \text{if is\_int? } x \text{ then } x + 7 \text{ else raise } (x + 1)$$
$$\text{in } \mathit{foo}\ (\lambda y.\, y)$$

# Contracts

# Contracts: prelude example

let $\mathit{double} = \lambda f.\, f\ (f\ 0)$ in

let $\mathit{pos} = \lambda i.\,$if $i < 0$ then false else true in

$\mathit{double}\ \mathit{pos}$

# Contracts: prelude example running

$$
\begin{aligned}
& \mathit{double}\ \mathit{pos} \\
\longrightarrow & \mathit{pos}\ (\mathit{pos}\ 0) \\
\longrightarrow^{*} & \mathit{pos}\ \mathsf{true} \\
\longrightarrow & \mathsf{if\ true} < 0\ \mathsf{then\ false\ else\ true} \\
\longrightarrow^{*} & \mathsf{Err}\ 1
\end{aligned}
$$

# Contracts: prelude example, defensive

```
let $double = \lambda f.$ if is_fun? $f$ then
  let $x = f\ 0$ in
  let $y = f$ (if is_int? $x$ then $x$ else raise 1) in
   if is_int? $y$ then $y$ else raise 1
    else raise 0 in
let $pos = \lambda i.$ if $i < 0$ then false else true in
$double\ pos$
```

# Contracts

$$
\begin{aligned}
e &::= \cdots \mid e_1 \longmapsto e_2 \mid \mathsf{monitor}(e_1, e_2) \\
v &::= \cdots \mid v_1 \longmapsto v_2 \mid \mathsf{monitor}(v, v_1 \longmapsto v_2) \\
E &::= \cdots \mid E \longmapsto e \mid v \longmapsto E \\
&\quad \mid \mathsf{monitor}(e, E) \mid \mathsf{monitor}(E, v)
\end{aligned}
$$

We also define a syntactic category to allow us to determine when a value is a function, or, a monitored function (with possibly many monitors wrapped around it.)

$$f ::= \lambda x.\, e \mid \mathsf{monitor}(f, v_1 \longmapsto v_2)$$

# Contracts

When we have a function application $f\ v'$ (where either $f$ is a function, or a monitored function), we first make sure that argument $v'$ passes contract $v_1$ (using $\text{monitor}(v', v_1)$), apply the function to the result, and make sure that the result of the function application will have contract $v_2$ called on it ($\text{monitor}((f\ (\text{monitor}(v', v_1))), v_2)$).

$$\frac{}{\begin{array}{c}(\text{monitor}(f, v_1 \longmapsto v_2))\ v' \longrightarrow \\ \text{monitor}((f\ (\text{monitor}(v', v_1))), v_2)\end{array}}$$

# Contracts

We also have rules to handle function application when the left operand is not a function or a monitored function, and also to handle flat contracts.

$$\frac{}{(\text{monitor}(v, v_1 \longmapsto v_2))\ v' \longrightarrow \text{raise } 3} *$$

$* \ v$ is not a function or a monitored function

$$\frac{}{\text{monitor}(v, v') \longrightarrow \text{if } v'\ v \text{ then } v \text{ else raise } 3} *$$

$$* \ v' \neq v_1 \longmapsto v_2$$

# Contracts: Error Propagation Rules

$$\frac{}{(\mathsf{Err}\ v \longmapsto e) \longrightarrow \mathsf{Err}\ v}$$

$$\frac{}{(e \longmapsto \mathsf{Err}\ v) \longrightarrow \mathsf{Err}\ v}$$

$$\frac{}{\mathsf{monitor}(\mathsf{Err}\ v, e) \longrightarrow \mathsf{Err}\ v}$$

$$\frac{}{\mathsf{monitor}(e, \mathsf{Err}\ v) \longrightarrow \mathsf{Err}\ v}$$

# Contracts: example

$$\begin{aligned}
&\text{let } \mathit{double} = \text{monitor}(\lambda f.\, f\ (f\ 0), \\
&\quad (\text{is\_int?} \longmapsto \text{is\_int?}\ ) \longmapsto \text{is\_int?}\ ) \text{ in} \\
&\text{let } \mathit{pos} = \text{monitor}(\lambda i.\, \text{if } i < 0 \text{ then false else true}, \\
&\quad \text{is\_int?} \longmapsto \text{is\_bool?}\ ) \text{ in} \\
&\mathit{double}\ \mathit{pos}
\end{aligned}$$