CS 3110

The Substitution Model

Nate Foster

Spring 2020

Today's scene: Cayuga Medical Center

Inductively defined relation

`e1 + e2` $\longrightarrow$ `e1' + e2`

if `e1` $\longrightarrow$ `e1'`

`v1 + e2` $\longrightarrow$ `v1 + e2'`

if `e2` $\longrightarrow$ `e2'`

`v1 + v2` $\longrightarrow$ `i`

if `i` *is the result of primitive operation* `v1 + v2`

Let semantics

```
let x = e1 in e2 → let x = e1' in e2
```

if `e1 → e1'`

```
let x = v1 in e2 → e2{v1/x}
```

Demo

Defining substitution: the easy parts

Nothing to do for integers:

$i \{v/x\} = i$

Just keep going through operations:

$(e1 + e2) \{v/x\} = (e1 \{v/x\}) + (e2 \{v/x\})$

Variables are where substitution really happens:

$x \{v/x\} = v$

$y \{v/x\} = y$

Defining substitution: let

(let y = e1 in e2) {v/x}

=

let y = (e1 {v/x}) in (e2 {v/x})

<u>Do</u> substitute in binding.
e.g.,
let x = 1 in
(let y = x in y)

(let x = e1 in e2) {v/x}

=

let x = (e1 {v/x}) in e2

<u>Stop</u> substituting in body.
e.g.,
let x = 1 in
(let x = 2 in x)

gets even trickier in the presence of functions: see textbook re. capture-avoiding substitution

If semantics

`if e1 then e2 else e3` $\longrightarrow$ `if e1' then e2 else e3`

    `if e1` $\longrightarrow$ `e1'`

`if true then e2 else e3` $\longrightarrow$ `e2`

`if false then e2 else e3` $\longrightarrow$ `e3`

Demo