CS 704

Aws Albarghouthi
PhD Toronto 2015

CS536 compilers

POPL

Does program P satisfy some property φ?

- → does it terminate
- → answer is correct
  functional correctness
- → memory-leak free
- → memory/CPU constraints
  $O(n^2)$
  constant-time
- → is it fair
- → crash-free
- → security/privacy
  information leaks
- → deadlock-freedom
- → data-race freedom

How do we enforce that P satisfies φ?

How can you check that P satisfies φ?

① designing PLs (type systems) that guarantee φ

② Static analysis
take a program P and analyze it to check if it satisfies φ

Research paper + presentation
40%

A number of assignments

website

canvas (just grades + submissions)

Systems / file system QP

Robotics / visualization

ML / ML code / verification for ML

Theory / complexity of certain verification

DB / datalog + static analysis

Optim /

Networks / QP that network administrators delete

λ calculus

a model of computation
Alonzo Church in 1920s

two constructs: function application
function definition

Turing machines $\equiv$ λ calculus

In mid 1960s, Peter Landin
"The next 700 programming languages"
"Correspondence between ALGOL 60 and Church's λ calculus"

McCarthy developed LISP
① simple, stripped functional programming language

Syntax of λ calculus

term t = x variable
| $\lambda x.\ t$ abstraction
| $t\ \ t$ application

two λ calculus terms

def f(x){
t
}

f(x)

f x

Given $\lambda x.t$
we say that x is bound in t

a variable that is not bound is free

identity combinator

$\lambda x. x$

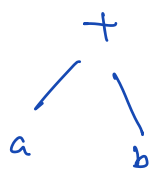
```
def f(x):
    return x
```

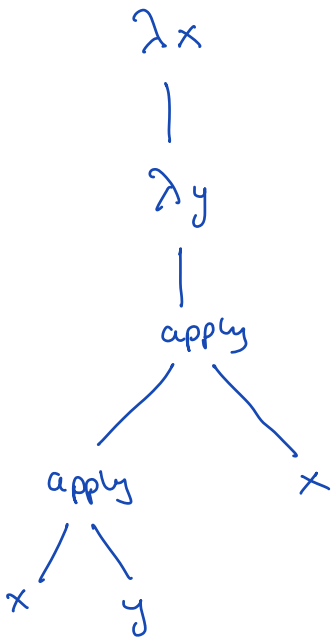
$\lambda x. \lambda y. x\ y\ x$

$\lambda x. (\lambda y. \underbrace{(x\ y)}\ \underbrace{x})$

$\lambda x. \lambda y. x\ (y x)$

$a + b$





x

$x \quad x$

$x \quad y$

$(x \quad y) \; x$

$(\lambda x . x) \; y$

Semantics

$1 + 2 \to 3$

$\lambda x . t$ is a one-argument function

$M \quad N$ apply M to N

$$(\lambda x . t) \; t' \longrightarrow t[x \mapsto t']$$

```
def f(x)
    return x+1
```

evaluate $f(10) \longrightarrow 10+1$

$$(\lambda x . x) \; y \longrightarrow y$$

$$(\lambda x.\ x\ (\lambda x.\ x))\ (u\ r)$$

$$\longrightarrow (u\ r)\ (\lambda x.\ x)$$

β-reduction

$$x\ (\lambda x.\ y)$$

$$(1+2) + (3+5)$$

$$\downarrow \qquad\qquad \downarrow$$