CS 444 - Compiler Construction **Winter 2020**

Lecture 16: March 4 , 2020

Lecturer: Ondřej Lhoták *Notes By: Harsh Mistry*

16.1 Static Analysis

- Analyse program to prove properties of its runtime behaviour
  - can program get stuck in infinite loop
  - Unreachable code
  - Division by 0
  - Dereference null
  - Access array out of bounds
  - Construct expression/computation.

Theorem 16.1 *Rice's Theorem: Let R be any non-trivial property of the execution of a program. Given a program l, it is undecideable whether P has property R*

Definition 16.2 *Non Trivial: $\exists P$ with property R and $\exists P'$ without property R*

Definition 16.3 *Analysis if conservative if its result is never untrue.*

Definition 16.4 *Analysis A is more precise then B if A gives a definitive answer for more programs*

- Java spec requires
  - Reachability analysis (Refer to JLS 14.20)
    - Every statement must potentially execute
    - No execution of a non-void method ends without a return statement
  - Define assignment (Refer to JLS 16)
    - Every local variable is written before it is read

16.1.1 Java Reachability

- For each statement S in program define
  - in[s] = can S start executing?
  - out[s] = can S stop executing?
- If in[s] = no for some S_i, then S is unreachable
- if out[s] = no for some S_i, then S is unreachable

- Constraints
  - L: if(E) S
    - in[S] = in[L]
    - out[L] = in[L] $\vee$ out[S]
  - L: if(E) S_1 else S_2
    - in[S_1] = in[L]
    - in[S_2] = in[L]
    - out[L] = out[S_1] $\vee$ out[S_2]
  - L: while(true) S
    - in[S] = in[L]
    - out[L] = no
  - L: while(false) S
    - in[S] = no
    - out[L] = in[L]
  - L: while(E) S
    - in[S] = in[L]
    - out[L] = in[L] ($\vee$ out[S])

While Constant Expression Note (JLS 15.28)

While loops have special static analysis checks to avoid infinite loops due to constant expression, but the definition of constant varies.

- `while(1 == 1) //constant`
- `while(x == x) //not constant`