Static Analysis

ThanhVu Nguyen
CSCE467

October 31, 2019

“Program testing can be used to show the presence of bugs, but never to show their absence.” (DJK, 1972)

What is Static Analysis?

Static Analysis

A method for automated reasoning on a representation of program

- **Static**: apply to some static representation (e.g., source code) of a program (in contrast to testing, profiling, or run-time checking)
- **Automated**: "push-button" technology, i.e., little user intervention

Applications

- **Compilers**: optimization (runtime, memory), remove dead code, etc
- **Verification**: verify program correctness

The Dream

Static Analyzer

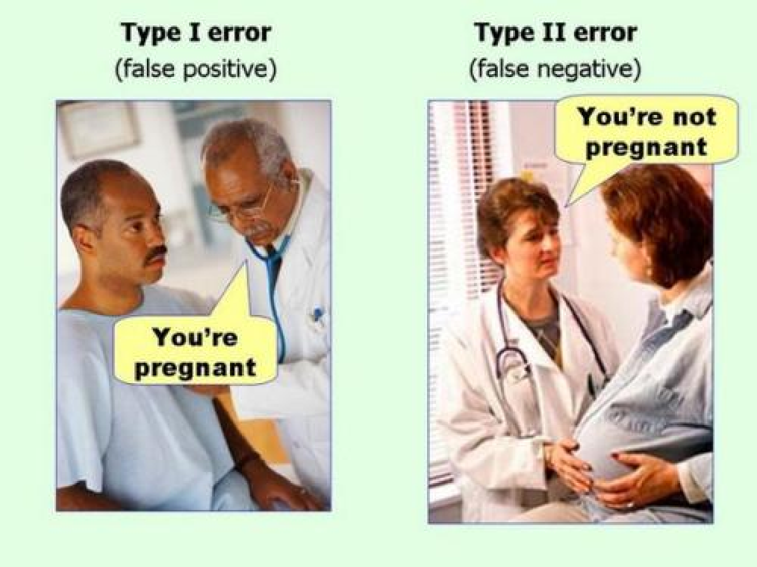
- **Inputs**: program, specifications (pre/post conditions, assertions)
- **Output**: correct/safe (provable), incorrect/unsafe (witness)

Requirements for a Perfect Analyzer

- **Soundness**: don't miss errors (no false negative)
- **Completeness**: don't raise false alarms (no false positive)
- **Termination**: always terminate

Question: is testing sound, complete, or terminate ?

False and True Positives



The Issue

Decision Problems

- Is the program P free of null ptr error?
- Does the program P satisfy given some given specification S?
- Does the program P terminate?

Rice Theorem (1953)

All non-trivial semantic questions about programs from a universal programming language are undecidable.

Approximation / Abstraction

- Example: $x = 42 \subseteq x \geq 40 \subseteq x \geq 0 \subseteq x \in Z$
- Approximate allows decidability and efficiency
- The approximation must still be *sound* , (often) sacrifice *completeness*, should preserve *termination*
- Properties:
  - **Precision**: must still be precise enough to give some *useful* answer
  - **Efficiency**: time/space usage
  - **Scalability**: work with realistic, real world programs

The WHILE language

Category	Domain	Meta variable
Numbers	$Z = \{0, 1, -1, \dots\}$	z
Truth values	$B = \{T, F\}$	t
Variables	$Var = \{x, y, ...\}$	x
Arithmetic expressions	AExp	a
Boolean expressions	BExp	b
Commands (statements)	Cmd	c

Context-Free Grammar of WHILE

$$\begin{aligned}
a ::= \quad & z \mid x \mid a1 + a2 \mid a1 - a2 \mid a1 * a2 \in \mathsf{AExp} \\
b ::= \quad & t \mid a1 = a2 \mid a1 > a2 \mid \neg b \mid b1 \wedge b2 \mid b1 \vee b2 \in \mathsf{BExp} \\
c ::= \quad & \mathsf{skip} \mid x := a \mid \\
& \mathsf{if}\ b\ \mathsf{then}\ c1\ \mathsf{else}\ c2\ \mathsf{end} \mid \\
& \mathsf{while}\ b\ \mathsf{do}\ c\ \mathsf{end} \in \mathsf{Cmd} \mid \\
& c1; c2
\end{aligned}$$

Example of a WHILE program

```
x := 6;
y := 7;
z := 0;
while x > 0 do
  x := x - 1;
  v := y;
  while v > 0 do
    v := v - 1;
    z := z + 1;
    end
end
```