COL874: Advanced Compiler Techniques

Modules 176-180

Sanyam Ahuja

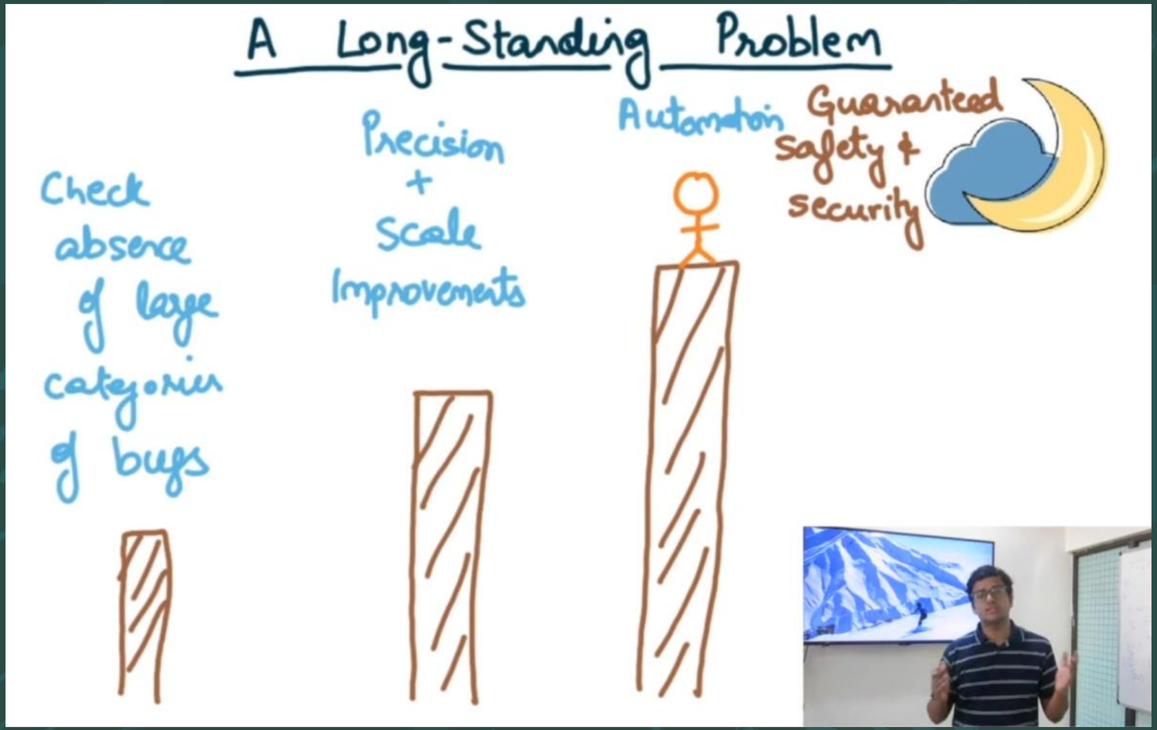
Recap

- Defined process level metrics for quality of program
- Critical applications require more stringent conditions
- Ariane V launcher’s failure - estimated cost of overflow
  - $500M direct cost
  - $2B indirect cost
- Most programs come without any warranty of any kind

Moving towards moon

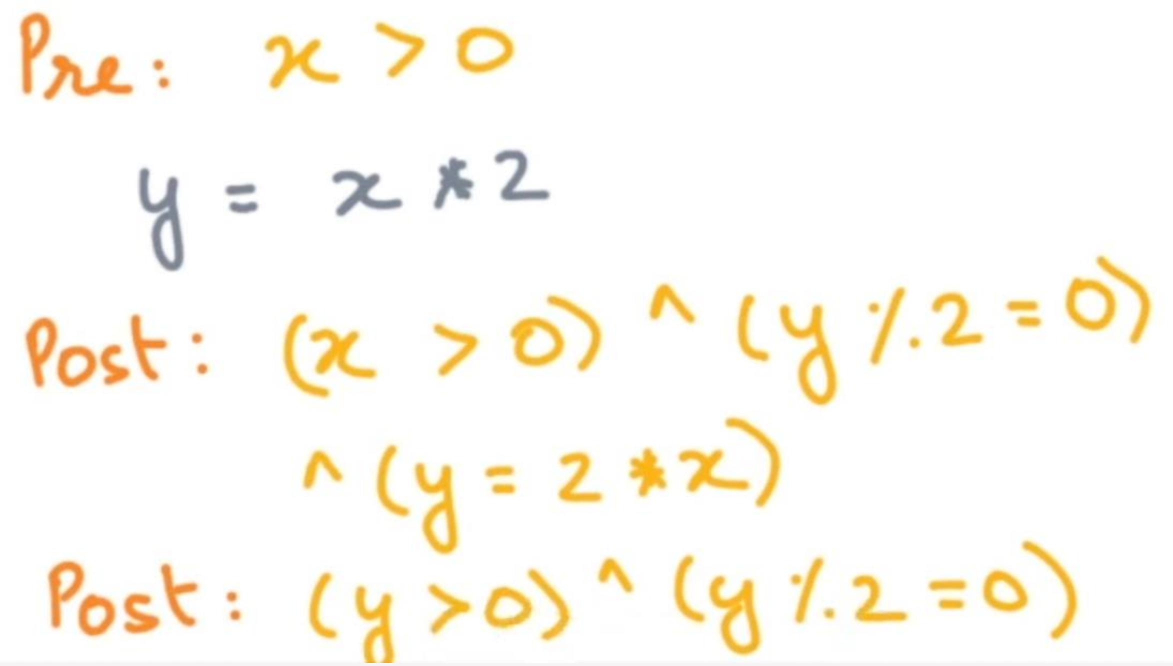
"The construction and application of a verifying compiler that guarantees the correctness of a program before running it"

- Tony Hoare, JACM 2003



Module 176 : Assertion

- A statement (logical predicate) about the values of the program variables at some program execution point.
- Precondition and Postcondition
    - Precondition : Assertion at program entry
    - Postcondition : Assertion at program exit



Partial Correctness

- If **precondition P** holds on entry of **program C** and program execution terminates, then **postcondition Q** holds, if and when the execution of C completes.
- Hoare Triple Notation
  - {P} C {Q}

Hoare Triple Notation Examples

Tautologies

- {P} C {true}
- {false} C {Q}

Non-terminating program C

- {P} C {false}
- {P} C {Q}

C : y := 2*x

- {true} C {y%2 = 0}
- {x=0} C {y=0}
- {x<10} C {y<20}
- {x<10} C {y%2=0}

Hoare Triple Notation Examples

```
C :    sum = 0;
       for(int i=0; i<n; i++) {
           sum += i;
       }
```

```
{true} C {sum = n(n-1)/2}
```

Abstraction

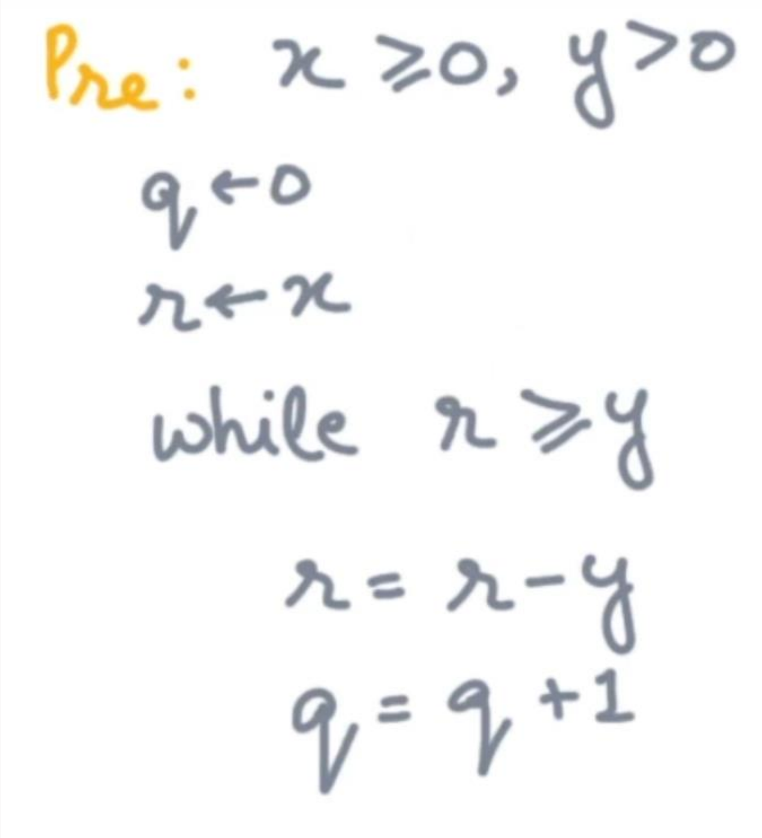
- An assertion that holds can be called an abstraction

```
y = 2,6,8,10,…
Assertion : {y>0 ^ y%2=0}
```

- An abstraction can add more behaviors, but not remove any

Module 177 : Invariants

- **Invariant** at a program point is an **assertion that holds** during execution whenever control reaches that point



Euclidean Integer Division Example

Pre: $x \geq 0, y > 0$

$q \leftarrow 0$ $x \geq 0 \quad y > 0$ $q = 0$

$r \leftarrow x$ " $\wedge\, r = x$

while $r \geq y$ $r \geq y$

     $r = r - y$ $r \geq 0$

     $q = q + 1$ $q \geq 0$

Post $0 \leq r < y, \quad q \geq 0$

Euclidean Integer Division Example

PROOF

Pre: $x \geq 0,\ y > 0$ — holds for base case — $x \geq 0\ y > 0$

$q \leftarrow 0$ — $q = 0$

$r \leftarrow x$ — " $\wedge\ r = x$

while $r \geq y$ — $r \geq y$

$r = r - y$

$q = q + 1$

assuming it holds at k^{th} iteration, it also holds for $(k+1)^{th}$ iteration — $r \geq 0$, $q \geq 0$

Post — $0 \leq r < y,\ q \geq 0$

Induction on the number of program steps. Also called inductive invariants

Euclidean Integer Division Example

Pre: $x \geq 0,\ y > 0$

$q \leftarrow 0$ —— $x \geq 0\ y > 0$, $q = 0$

$r \leftarrow x$ —— " $\wedge r = x$

while $r \geq y$ —— $r \geq y$, $r \geq 0$, $q \geq 0$

$r = r - y$

$q = q + 1$

Post —— $0 \leq r < y,\ q \geq 0$

$x = qy + r$

Euclidean Integer Division Example

Pre: $x \geq 0, \; y > 0$

$x \geq 0 \;\; y > 0$

$q \leftarrow 0$ —————— $q = 0$

$r \leftarrow x$ —————— " $\wedge \; r = x$

while $r \geq y$ —————— $r \geq y$

$r \geq 0$

$q \geq 0$

$x = qy + r$

$r = r - y$

$q = q + 1$

Post —————— $0 \leq r < y, \;\; q \geq 0$

$x = qy + r$.

Module 178 : Verification Conditions

Verification Conditions

Pre: $x \geqslant 0, y > 0$

$q = 0$
$r = x$

while $r \geqslant y$

$r = r - y$
$q = q + 1$

$x = qy + r$
$r \geqslant y$

Post: $x = qy + r$
$q, r \geqslant 0$
$r < y$

Verification Conditions

Verification Condition 1(VC1)

```
{x>=0, y>0}
    q=0; r=x;
    if(r>=y)
{x=qy+r, r>=y}
```

Verification Condition 2(VC2)

```
{x=qy+r, r>=y}
    r'=r-y; q'=q+1;
    if(r'>=y)
{x=q'y+r', r'>=y}
```

Verification Condition 3(VC3)

```
{x=qy+r, r>=y}
    r'=r-y; q'=q+1;
    if not(r'>=y)
{x=q'y+r', r'<y}
```

Verification Conditions

Precondition:
```
{x>=0, y>0}
```

C:
```
q=0;
r=x;
while(r>=y)
    r=r-y;
    q=q+1;
```

Postcondition:
```
{x=qy+r, q,r>=0, r<y}
```

VC1 ^ VC2 ^ VC3

```
{x>=0, y>0}
C
{x=qy+r, r<y, q,r>=0}
```

Assignment Verification Condition

Precondition: `{P(X,Y,…)}`
C: `X := E(X,Y,…)`
Postcondition: `{Q(X,Y,…)}`

$\forall\ X,Y,\ldots\ :\ (\exists X' : P(X',Y,\ldots)\ \wedge\ X=E(X',Y,\ldots))\ \Rightarrow\ Q(X,Y,\ldots)$ (Floyd)

$\forall\ X,Y,\ldots\ :\ P(X,Y,\ldots)\ \Rightarrow\ Q(X,Y,\ldots)[X:=E]$ (Hoare)

`B[x:=A]` represents substitution of A for x in B

Assignment VC Example

Precondition: {X>=0}
C: X := X+1
Postcondition: {X>0}

$\forall$ X : ($\exists$X′:X′>=0 ^ X=X′+1) $\Rightarrow$ X>0 (Floyd)

$\forall$ X : X>=0 $\Rightarrow$ X+1>0 (Hoare)

Module 179: Conditional Verification Condition

```
{P(X,…)}
if B(X,…) then
     {P1(X,…)}          P(X,…)^B(X,…)  ⇒  P1(X,…)
     …                  (VC1)
     {P2(X,…)}
else
     {P3(X,…)}          P(X,…)^¬B(X,…)  ⇒  P3(X,…)
     …                  (VC3)
     {P4(X,…)}
fi
{Q(X,…)}                P2(X,…)VP4(X,…)  ⇒  Q(X,…)
                        (VC5)
```

Conditional VC Example

```
{X=x_0}
if X>=0 then
    {X=x_0^X>=0}
    skip
    {X=x_0^X>=0}
else
    {X=x_0^X<0}
    X := -X
    {X=-x_0^X>0}
{X=|x_0|}
```

$(X=x_0)\wedge(X>=0) \Rightarrow (X=x_0 \wedge X>=0)$ (VC1)

$(X=x_0)\wedge(X>=0) \Rightarrow (X=x_0 \wedge X>=0)$ (VC2)

$(X=x_0)\wedge\neg(X>=0) \Rightarrow (X=x_0 \wedge X<0)$ (VC3)

$(X=x_0 \wedge X<0) \Rightarrow (-X=-x_0 \wedge -X>0)$ (VC4)

$(X=x_0 \wedge X>=0)\vee(X=-x_0 \wedge X>0) \Rightarrow X=|x_0|$ (VC5)

Module 180: Sequence Operator Verification Condition

$\{P(X,Y,\ldots)\}$
$X{:=}f(X,Y,\ldots)$
$\{P_1(X,Y,\ldots)\}$
$Y{:=}g(X,Y,\ldots)$
$\{Q(X,Y,\ldots)\}$

$\forall\ X,Y,\ldots\ :\ P(X,Y,\ldots)$
$\Rightarrow\ P_1(X,Y,\ldots)[X{:=}f]$

$\forall\ X,Y,\ldots\ :\ P_1(X,Y,\ldots)$
$\Rightarrow\ Q(X,Y,\ldots)[Y{:=}g]$

$\{P\}\quad s_1\quad \{P_1\}$
$\{P_1\}\quad s_2\quad \{Q\}$

Choose $P_1(X,Y,\ldots)=Q(X,Y,\ldots)[Y{:=}g]$

Module 180: Sequence Operator Verification Condition

{P(X,Y,…)}
X:=f(X,Y,…)
{Q(X,Y,…)[Y:=g]}
Y:=g(X,Y,…)
{Q(X,Y,…)}

∀ X,Y,… : P(X,Y,…)
⇒Q(X,Y,…)[Y:=g][X:=f]

∀ X,Y,… : P(X,Y,…)⇒Q(X,Y,…)[Y:=g][X:=f]

Thank You