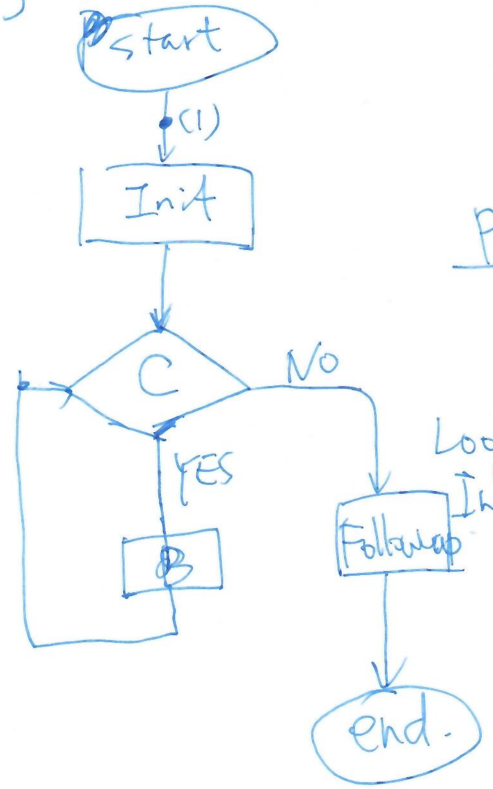
(2018-03-10 ~ 2018-03-12) 2-1: Correctness of Algorithms

hfwei@nju.edu.cn

Alg:



Weakest ↓ . Strongest ↑.

Goal: P {Alg} Q.

(Meaning: P & Alg ⇒ Q).

Pf:

Partial Correctness:

Loop Invariant:

(1) P {Init} I. (Basic Step)

(2) (I ∧ C) {B} I (Inductive Step).

(3) (I ∧ ¬C) {Follow-up} Q.

Termination (不讲 loop variant convergence).

X: state of alg.

D(X): monotonically decreasing function.

Well-ordering set (ℕ).

(1) (I ∧ C) {B} (D(X') < D(X))

Q₂: [illegible]

(2) (I ∧ D(X) = min) ⇒ ¬C.

引入

Method / Procedure ()

// (1) P: pre-condition.

Init

Q₁: I 在(2)处还是在(2)'处?

// (2) I: loop invariant.

while (C) (2)' ← (I ∧ C)

B

// (3) After the loop [I ∧ ¬C]

Followup

// Q: post-condition.

Alg.

Leibniz: Let Us Calculate!

Q: 循环语句中有分支(if)怎么办: (blackbox)

Q: 变量一直在变, 初始已弃了怎么办?

How to Apply Hoare Logic? (Let Us Calculate)

(1) For Proof (Discussed Today).

- Given P, Q, Alg.
- Goal: P {Alg} Q. (Key: loop invariant).

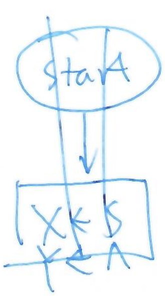
(2) For design.

- Given P, Q.
- Goal: design Alg (~~including~~ developing loop & invariants / loop.
  代码模板

How to develop loop invariant:

$$I \triangleq (totalWork = workDone + workToDo).$$

EX.



说 Reverse(S):

```
X ← S
Y ← Λ
while (X ≠ Λ)
    Y ← head(X)·Y
    X ← tail(X)
return Y.
```

$I \triangleq$ (reverse(S) = reverse(X) · Y)

vs. S = reverse(Y) · X.

X = Φ.

Y = reverse(S)

~~equal(X, Y)~~

equal(S_1, S_2) (problem 5.9).

- head(X)
- tail(X)
- last(X)
- all-but-last(X).
- eq(s,t)

// (0) P: S_1, S_2 are strings. $P = T$

$X \leftarrow S_1$

$Y \leftarrow S_2$

$E \leftarrow T$

While $X \neq \Lambda \wedge Y \neq \Lambda \wedge E == T$

if eq(head(X), head(Y))

$X \leftarrow$ tail(X)

$Y \leftarrow$ tail(Y)

else

$E \leftarrow \perp$

return ~~$\perp$~~

// (2)

// (3)

else return T. // (4)

// (5)

Total work ~~TODO~~ workDone

(1) I: $S_1 = S_2 \Leftrightarrow X = Y$ ~~$\wedge E = T$~~

Q: $S_1 = S_2 \Leftrightarrow X = Y$. a loop invariant?

IS

(2) $I \wedge (X = \Lambda \vee Y = \Lambda \vee E = \perp)$.

$S_1 = S_2 \Leftrightarrow (X = Y \wedge E = T) \wedge (X = \Lambda \vee Y = \Lambda \vee E = \perp)$

Simplify it:

$S_1 = S_2 \Leftrightarrow (X = \Lambda \wedge Y = \Lambda) \wedge E = T$

(3) $S_1 = S_2 \Leftrightarrow (X = \Lambda \wedge Y = \Lambda \wedge E = T) \wedge \neg(X = \Lambda \wedge Y = \Lambda) \wedge (E =$

$\Rightarrow$ ~~$S_1 = S_2$~~ $S_1 \neq S_2 \wedge E = \perp$.

(5) $S_1 = S_2 \Leftrightarrow E = T$.

(4) the empty "else".

$S_1 = S_2 \Leftrightarrow X \neq \Lambda \wedge Y = \Lambda \wedge E = T$

$X = \Lambda \wedge Y = \Lambda$

$\Rightarrow S_1 = S_2 \Leftrightarrow E = T$.

(5) $\Rightarrow$ (4).

Q: How to deal with "if"?

Look back "Equal(S_1, S_2):

①本身是不变式吗?
②这个不变式够用吗?

(1) $I \triangleq (S_1 = S_2 \Leftrightarrow X = Y)$

(2): $S_1 = S_2 \Leftrightarrow X = Y \wedge (X = \Lambda \vee Y = \Lambda \vee E = \perp)$

~~$\Leftrightarrow (X = \Lambda \wedge Y = \Lambda) \vee (X = Y \wedge E = \perp)$.~~

~~$\Leftrightarrow X = \Lambda \wedge Y = \Lambda$~~

~~$\Leftrightarrow (X = \Lambda \wedge Y = \Lambda) \vee E = \perp$.~~

$\Leftrightarrow (X = \Lambda \wedge Y = \Lambda) \vee (X = Y \wedge E = \perp)$. ×

再看例子 Problem 5.12. Pal(S) 見下页.