研究生课程

程序验证方法
Rely-Guarantee Method

朱惠彪

华东师范大学软件学院

Related Paper:

- Qiwen Xu, Willem P. de Roever, Jifeng He:
  The Rely-Guarantee Method for Verifying Shared Variable Concurrent Programs. Formal Asp. Comput. 9(2): 149-174 (1997)

- **Owicki and Gries Method:**

an interference freedom test

formulated in such a way that the knowledge of the complete system code is assumed

- **Rely-Guarantee Method:**

The key point to achieve compositionality is to reformulate this interference freedom test.

The Language Syntax

- P ::= $\bar{x} \coloneqq \bar{e}$ | $P_1; P_2$ | $\boldsymbol{if}\ b_1 \rightarrow P_1 \square \ldots \square\ b_n \rightarrow Pn\ \boldsymbol{fi}$ | $\boldsymbol{while}\ b\ \boldsymbol{do}\ P\ \boldsymbol{od}$ | $\boldsymbol{await}\ b\ \boldsymbol{then}\ P\ \boldsymbol{end}$ | $P_1 \mid\mid P_2$

A Proof System for Partial Correctness

- **Example (Owicki & Gries Method)**

$$P_1 :: x := x + 1 \,\|\, P_2 :: x := x + 2$$

Let $p_1 \equiv x = 0 \lor x = 2 \qquad q_1 \equiv x = 1 \lor x = 3$

$\qquad p_2 \equiv x = 0 \lor x = 1 \qquad q_2 \equiv x = 2 \lor x = 3$

The local verification:

$$\underset{r}{\underline{\{p_1\}}}\ \overset{R}{x := x + 1}\ \underset{r}{\underline{\{q_1\}}}$$

$$\underset{r}{\underline{\{p_2\}}}\ \underset{R}{x := x + 2}\ \underset{r}{\underline{\{q_2\}}}$$

A Proof System for Partial Correctness

Interference freedom test: four cases.

$$\{p_1 \wedge p_2\}\ x := x + 2\ \{p_1\}$$

$$\{q_1 \wedge p_2\}\ x := x + 2\ \{q_1\}$$

$$\{p_1 \wedge p_2\}\ x := x + 1\ \{p_2\}$$

$$\{p_1 \wedge q_2\}\ x := x + 1\ \{q_2\}$$

Satisfied!

Specification

$$(pre, rely, guar, post)$$

- The **assumption** is composed of *pre* and *rely*.
- The **commitment** is composed of *guar* and *post*.
- *P* <u>sat</u> (*pre*, *rely*, *guar*, *post*), if

  1) *P* is invoked in a state which satisfies *pre*, and

  2) any environment transition satisfies *rely*,

  then

  3) any component transition satisfies *guar*,

  4) if a computation terminates, the final state satisfies *post*.

$$x := 10\ \underline{sat}\ (true, x > 0 \rightarrow x' \geq x, true, x \geq 10)$$

Proof System

- **Some notations:**

For two predicates $f(y, y_0)$ and g(y, y', y_0), let f *stable when* g be a shorthand for $\forall y, y', y_0.\ f(y, y_0) \wedge g(y, y', y_0) \rightarrow f(y', y_0)$ and $f'(y, y_0)$ denote $f(y', y_0)$.

Proof System

- Assignment axiom

$$\frac{\begin{array}{c} pre \longrightarrow post[\bar{e}/\bar{x}] \\ (pre \wedge [\bar{x}' = \bar{e}]) \longrightarrow guar \\ pre\ \underline{stable\ when}\ rely \\ post\ \underline{stable\ when}\ rely \end{array}}{\bar{x} := \bar{e}\ \underline{sat}\ (pre, rely, guar, post)}$$

- $[\bar{x}' = \bar{e}] \stackrel{\text{def}}{=} (\bar{x}' = \bar{e} \vee x' = x) \wedge \forall z \in (y - \bar{x}).\ z' = z.$
- In a typical computation, there are a number of environment transitions before and after the component transition. Since *pre* holds initially, it follows from ***pre*** ***<u>stable when</u>*** ***rely*** that *pre* still holds immediately before the component transition.
- Due to ***post*** ***<u>stable when</u>*** ***rely***, ***post holds*** ***in any states after a number of environment transitions***.
- $x := 10\ \underline{sat}\ (true, x > 0 \rightarrow x' \geq x, true, x \geq 10)$

Proof System

- Consequence rule

$$\frac{pre \rightarrow pre_1, rely \rightarrow rely_1,\ guar_1 \rightarrow guar,\ post_1 \rightarrow post \qquad P\ sat\ (pre_1,\ rely_1,\ guar_1\ ,\ post_1)}{P\ \underline{sat}\ (pre,\ rely,\ guar,\ post)}$$

- $\boldsymbol{x := 10\ \underline{sat}\ (x = -2, x > 0 \rightarrow x' \geq x, true, x \geq 10 \lor x = -6)}$

Proof System

- Sequential composition rule

$$\frac{\begin{array}{c} P\ sat\ (pre, rely, guar, mid) \\ Q\ \underline{sat}\ (mid, rely, guar, post) \end{array}}{P;Q\ \underline{sat}\ (pre, rely, guar, post)}$$

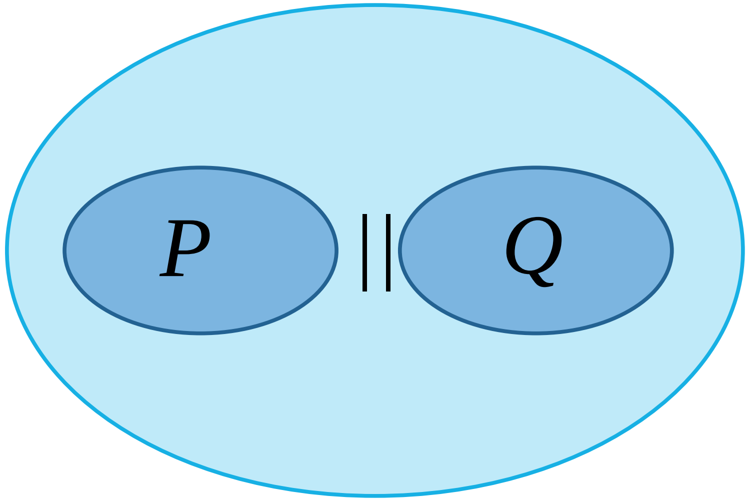
- $$\frac{\begin{array}{l} \boldsymbol{x := x + 1\ \underline{sat}\ (x \geq x_0, x_0 \leq x \rightarrow x \leq x', x' \geq x, x \geq x_0 + 1)} \\ \boldsymbol{x := x + 1\ \underline{sat}\ (x \geq x_0 + 1, x_0 \leq x \rightarrow x \leq x', x' \geq x, x \geq x_0 + 2)} \end{array}}{\boldsymbol{x := x + 1; x := x + 1\ \underline{sat}\ (x \geq x_0, x_0 \leq x \rightarrow x \leq x', x' \geq x, x \geq x_0 + 2)}}$$

Proof System

- Parallel rule

$$\frac{\begin{array}{c}(rely \vee guar_1) \rightarrow rely_2 \\ (rely \vee guar_2) \rightarrow rely_1 \\ (guar_1 \vee guar_2) \rightarrow guar \\ P\ \underline{sat}\ (pre, rely_1, guar_1, post_1) \\ Q\ \underline{sat}\ (pre, rely_2, guar_2, post_2)\end{array}}{P \parallel Q\ \underline{sat}\ (pre, rely, guar, post_1 \wedge post_2)}$$

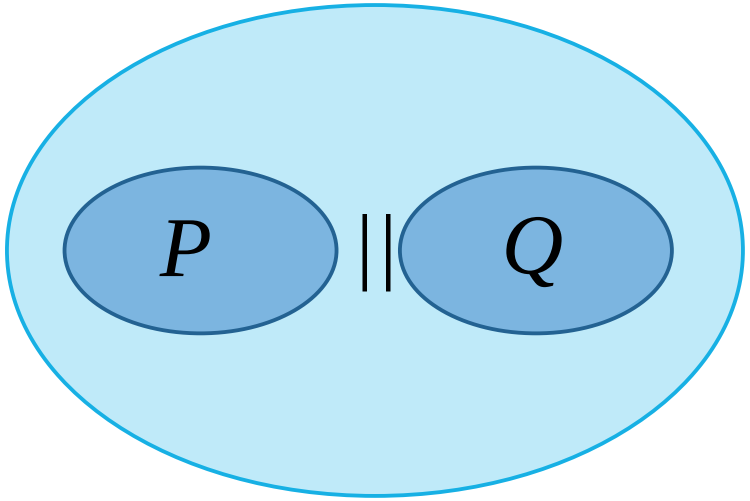
Proof System



$$\frac{\begin{array}{c}(rely \vee guar_1) \to rely_2 \\ (rely \vee guar_2) \to rely_1 \\ (guar_1 \vee guar_2) \to guar \\ P\ \underline{sat}\ (pre, rely_1, guar_1, post_1) \\ Q\ \underline{sat}\ (pre, rely_2, guar_2, post_2)\end{array}}{P \mid\mid Q\ \underline{sat}\ (pre, rely, guar, post_1 \wedge post_2)}$$

- **Assume the overall environment is *R*. The environment of process *P* consists of *Q* and *R*, and the environment of process *Q* consists of *P* and *R*.**
- **The strongest rely-condition that *P* can assume is $rely \vee guar_2$ and the strongest rely-condition for *Q* is $rely \vee guar_1$.**

Proof System



$$\frac{\begin{array}{c}(rely \vee guar_1) \rightarrow rely_2 \\ (rely \vee guar_2) \rightarrow rely_1 \\ (guar_1 \vee guar_2) \rightarrow guar \\ P\ \underline{sat}\ (pre, rely_1, guar_1, post_1) \\ Q\ \underline{sat}\ (pre, rely_2, guar_2, post_2)\end{array}}{P \mid\mid Q\ \underline{sat}\ (pre, rely, guar, post_1 \wedge post_2)}$$

- **A component transition of *P* || *Q* is either from *P* or from *Q*, and hence it satisfies $guar_1 \vee guar_2$.**
- **When both *P* and *Q* have terminated, from the two sub-specifications it follows that both $post_1$ and $post_2$ are satisfied.**

Thanks!