Verification and validation: Quick reference

vishvAs vAsuki

December 12, 2009

Mainly from Alan Cline's handouts.

Part I
Notation

S, S_i, $S_i;S_j$: Pieces of code.

Part II
Themes

1 Validation

Are we trying to make the right thing? Are we solving the right equation?

2 Verification

Have we made what we were trying to make? Are there errors in our implementation of the model?

Part III
Reasoning about programs

Also see distributed computing ref.

3 Correctness

3.1 Assertion

Assertion p: Assume that p is true.

In practice, good programmers use special statements to check if assertions are true; eg: Java.

3.2 Correctness

S is correct wrt precondition p and postcondition q if: starting with p true, run S, get q.

3.3 Partial correctness

$\{p\}\, S\, \{q\}$. S is correct wrt p and q if: starting with p true, if ye run S, ye get q if S terminates. This is useful notation for proving correctness of program segments.

3.3.1 Axioms

[Hoare] F indicates set of empty set of states (unreachable); so:
$\forall q, S :: F S\, \{q\}$; $\{p\}\, S F \implies \neg p$: if S results in unreachable state, initial state itself must have been unreachable.

$\{p_1\}\, S_1\, \{p_2\} \wedge \{p_2\}\, S_2\, \{p_3\} \implies \{p_1\}\, S_1; S_2\, \{p_3\}$.

4 Verification with forward chaining

4.1 Picking invariants

During verification, select invariant weak enough to remain true before and after loop is executed, also strong enough to lead to the required postcondition: necessary to ensure postcondition even if loop not entered.

4.2 Translate program into hoare triples

If S = if cond then S_1 else S_2: $(\{p \wedge cond\}\, S_1\, \{q\}) \wedge (\{p \wedge \neg cond\}\, S_2\, \{q\})$.

Iteration: S = while cond do S', $q = (p \wedge \neg cond)$: $(\{p \wedge cond\}\, S\, \{p\})$: p is the loop invariant; cond is loop variant. p can be false during loop execution, but returns to true in the end.

Assignment: $\{p(x)\}\, x := E\, \{p(E)\}$.

5 Verification with preconditions

Aka back substitution. This is backward chaining.

5.1 Weakest preconditions for program S, postcondition q

p = wp(S,q). Weakest assertion p: $\{p\}\, S\, \{q\}$. For any r : if $\{r\}\, S\, \{q\}$ $\wedge$ S terminates; $r \implies wp(S,q)$. Converse is true.

So, use this if you want to show that $\{r\}\, S\, \{q\}$ (like $\{r\}\, x := 5\, \{x \geq 5\}$): take q, substitute the effects of S in q, thence get wp(S, q); show $r \implies wp(S,q)$!

$wp(S_1; S_2, q) = wp(S_1, wp(S_2, q))$.

wp(if cond then S; q) = $(cond \implies wp(S,q)) \wedge (\neg cond \implies q)$.

$wp(x := E, q(x))$ = E is defined, q(E) true.