

# Program Semantics; Hoare Triples

## CS 536 Lecture 6, Mon Jan 30, 2012

### A. Why

- The meaning of a program is that it transforms states.
- To specify a program's correctness, we need to know its precondition (what must be true before executing it) and its postcondition (what should be true after it).

### B. Objectives

- At the end of today you should
- Know the basic semantics of our programming language.
- Know the semantics of our programs at intuitive and formal levels.
- Know the syntax of correctness triples (a.k.a. Hoare triples).
- Know what it means for a program to satisfy its specification.

### C. The Meaning of Programs as State Transformers (continued)

- Last week we saw that the meaning of a program is that it transforms the state:
  - $M(S, \sigma) = \tau$  where  $S$  is a statement,  $\sigma$  is the current state, and  $\tau$  is the new state.
  - Simple assignment:  $M(v := e, \sigma) = \sigma[v \mapsto \sigma(e)]$ 
    - Update the left-hand-side variable with the value of the right-hand side.
  - Array assignment:  $M(b[e_1] := e_2, \sigma) = \sigma[b \mapsto \sigma(b)][\sigma(e_1) \mapsto \sigma(e_2)]$ 
    - Update the array element with the given index by the value of the right-hand side.
  - Sequence:  $M(S_1 ; S_2, \sigma) = M(S_2, M(S_1, \sigma))$ 
    - Run  $S_1$  in the current state and then run  $S_2$  in the state that results.
  - Conditional: If  $S \equiv \mathbf{if } B \mathbf{ then } S_1 \mathbf{ else } S_2 \mathbf{ fi}$ , then
    - $M(S, \sigma) = M(S_1, \sigma)$  if  $\sigma \models B$ ;  $M(S, \sigma) = M(S_2, \sigma)$  if  $\sigma \models \neg B$ .
- Continuing today, for the iterative statement, let  $W \equiv \mathbf{while } B \mathbf{ do } S \mathbf{ od}$ , then
  - $M(W, \sigma) = \sigma$  if  $\sigma \models \neg B$ .
    - $W$  behaves like **skip** if the test fails.
  - $M(W, \sigma) = M(W, M(S, \sigma))$  if  $\sigma \models B$ .
    - If the test is true, run the loop body followed by the rest of the loop.

- Essentially, look at the sequence of states  $\sigma$ ,  $M(S, \sigma)$ ,  $M(S; S, \sigma)$ ,  $M(S; S; S, \sigma)$ , .... If all of them satisfy  $B$ , then  $M(W, \sigma)$  is undefined. Otherwise,  $M(W, \sigma)$  is the first state that satisfies  $\neg B$ .
- If it helps, we can give names to the states in the sequence:
  - Let  $\tau_0 = \sigma$ , let  $\tau_1 = M(S, \tau_0)$ , ...,  $\tau_{j+1} = M(S, \tau_j)$ , ....
  - Then  $M(W, \sigma) = \tau_k$  where  $\tau_k \models \neg B$  and  $\tau_0, \dots, \tau_{k-1}$  all  $\models B$ , if such a  $k$  exists.
    - $M(W, \sigma)$  is undefined if no such  $k$  exists.
- **Loop Example 1:**
  - Let  $W \equiv \mathbf{while} \ x \leq n \ \mathbf{do} \ S \ \mathbf{od}$ , where  $S \equiv x := x+1; y := y+y$ .
  - Let
    - $\sigma = \tau_0 = \{x = 0, n = 2, y = 1\}$
    - $\tau_1 = M(S, \tau_0) = \{x = 1, n = 2, y = 2\}$
    - $\tau_2 = M(S, \tau_1) = \{x = 2, n = 2, y = 4\}$ , and
    - $\tau_3 = M(S, \tau_2) = \{x = 3, n = 2, y = 8\}$ .
  - Since  $\tau_0, \tau_1$ , and  $\tau_2$  all  $\models x \leq n$  and  $\tau_3 \models x > n$ , we find  $M(W, \sigma) = \tau_3$ .
- **Loop Example 2:** Let  $W \equiv \mathbf{while} \ x \neq n \ \mathbf{do} \ x := x-1 \ \mathbf{od}$ .
  - Let
    - $\sigma = \tau_0 = \{x = 2, n = 0\}$
    - $\tau_1 = M(S, \tau_0) = \{x = 1, n = 0\}$
    - $\tau_2 = M(S, \tau_1) = \{x = 0, n = 0\}$
  - Since  $\tau_0$  and  $\tau_1$  both  $\models x \neq n$  and  $\tau_2 \models x = n$ , we have  $M(W, \sigma) = \tau_2$ .
- **Loop Example 3:** Again, let  $W \equiv \mathbf{while} \ x \neq n \ \mathbf{do} \ x := x-1 \ \mathbf{od}$ .
  - This time let
    - $\sigma = \tau_0 = \{x = -1, n = 0\}$
    - $\tau_1 = \{x = -2, n = 0\}$
    - $\tau_2 = \{x = -3, n = 0\}$
    - ....
    - $\tau_j = M(S, \tau_{j-1}) = \{x = -j-1, n = 0\}$
  - Since all of these states satisfy  $x \neq n$ , we have that  $M(W, \sigma)$  is undefined.

### D. Correctness Triples (“Hoare Triples”)

- A **correctness triple** (a.k.a. “**Hoare triple**,” after C.A.R. Hoare) is a program  $S$  plus its specification predicates  $p$  and  $q$ .

- The **precondition**  $p$  describes what we're assuming is true about the state before the program begins.
- The **postcondition**  $q$  describes what should be true about the state after the program terminates.
- **Syntax of correctness triples:**  $\{p\} S \{q\}$  (Think of it as `/* p */ S /* q */`)
  - **Note !!!** the braces are not part of the precondition or postcondition. Saying "The precondition is  $\{p\}$ " is like saying "The test for `if B then S fi` is `if B then.`"
  - The precondition of  $\{p\} S \{q\}$  is  $p$ , not  $\{p\}$ . Similarly the postcondition is  $q$ , not  $\{q\}$ .
- **Informal semantics; Total and Partial Correctness**
  - Informally, for a state to **satisfy**  $\{p\} S \{q\}$ , it must be that if the state satisfies  $p$ , then after running  $S$ ,  $q$  is satisfied.
  - But we have to distinguish between programs that **terminate** (i.e., stop without error) and programs that **diverge** (i.e., run forever).
  - **Partial correctness:** We require that **if**  $S$  terminates under  $\sigma$ , then it terminates in a state satisfying  $q$ . If  $S$  diverges under  $\sigma$ , the triple is still partially correct.
  - **Total correctness:** We require that  $S$  must terminate, and terminate in a state satisfying  $q$ . If  $S$  diverges under  $\sigma$ , the triple is not totally correct.
  - The reason for distinguishing between partial and total correctness is that we can often separate proofs of partial correctness from proofs of termination.
  - A program-with-specification  $\{p\} S \{q\}$  is valid if it is satisfied in all states.
    - If it is not satisfied in some state, then the program has a bug in that state.

## E. Satisfaction and Validity of a Correctness Triple

- Analogous to  $\sigma \models p$  and  $\models p$  for satisfaction and validity of predicates, we'll use the notation  $\sigma \models \{p\} S \{q\}$  to say that  $\sigma$  **satisfies the triple**, and we'll write  $\models \{p\} S \{q\}$  to say that the triple is valid (satisfied in all states).
- However, we'll distinguish between  $\models$  for partial correctness and  $\models_{tot}$  for total correctness.
- **Partial correctness:**
  - For the triple  $\{p\} S \{q\}$  to be satisfied in a state under partial correctness, if the precondition is satisfied and running  $S$  terminates, then the terminating state must satisfy the postcondition.
  - $\sigma \models \{p\} S \{q\}$  means that if  $\sigma \models p$  **and**  $M(S, \sigma)$  is defined, **then**  $M(S, \sigma) \models q$ .
- **Total correctness**
  - For the triple  $\{p\} S \{q\}$  to be satisfied in a state under total correctness, if the precondition is satisfied, then running  $S$  must terminate, in a state that satisfies the postcondition.

- $\sigma \models_{tot} \{p\} S \{q\}$  means that if  $\sigma \models p$  **then**  $M(S, \sigma)$  is defined **and**  $M(S, \sigma) \models q$ .
- What if  $p$  is not satisfied? Will  $S$  terminate? Will  $q$  be satisfied?
- **Example 1:**  $\{x > 0\} x := x+1 \{x > 0\}$ . For both kinds of correctness, this is satisfied in all states, so the triple is valid.
- **Example 2:**  $\{x > 0\} x := x-1 \{x > 0\}$ . For both kinds of correctness, this is satisfied in every state where  $x \neq 1$ . If (we're in a state that satisfies)  $x = 1$ , then the triple is not satisfied. This triple is not valid.
- **Example 3:**  $\{k \geq 0\} W \{k = 0\}$  where  $W \equiv \mathbf{while} \ k \neq 0 \ \mathbf{do} \ k := k-1 \ \mathbf{od}$ 
  - This is valid under both partial and total correctness: If  $\sigma \models k \geq 0$  then the loop will terminate (in  $\sigma(k)$  steps, in fact) in a state where the postcondition  $k = 0$  is satisfied.
- **Example 4:**  $\{\mathbf{true}\} W \{k = 0\}$  where  $W$  is the same loop as in Example 3.
  - This triple is still valid under partially correctness but it's no longer valid under total correctness.
  - **Partial correctness:**
    - $\sigma \models \{\mathbf{true}\} W \{k = 0\}$  iff  $((\sigma \models \mathbf{true} \text{ and } M(W, \sigma) \text{ defined}) \text{ implies } M(W, \sigma) \models k=0)$ .
    - Since  $\sigma$  always  $\models \mathbf{true}$ , this turns into  $(M(W, \sigma) \text{ defined implies } M(W, \sigma) \models k=0)$ , which holds for all  $\sigma$ .
  - **Total correctness:**
    - $\sigma \models_{tot} \{\mathbf{true}\} W \{k = 0\}$  is equivalent to
    - $(\sigma \models \mathbf{true} \text{ implies } (M(W, \sigma) \text{ defined and } M(W, \sigma) \models k = 0))$ , which is equivalent to
    - $M(W, \sigma)$  is defined and  $M(W, \sigma) \models k = 0$
    - But  $M(W, \sigma)$  isn't defined if  $\sigma \models k < 0$ . (For any  $\sigma \models k \geq 0$ , it *is* defined and  $\models k = 0$ .)
    - So  $\{\mathbf{true}\} W \{k = 0\}$  is not totally correct.

## F. Some Common Questions about Correctness Triples

- The definition of satisfaction for  $\{p\} S \{q\}$  isn't trivial, and it's worth looking at different cases that can occur.
- *What if the precondition  $p$  isn't satisfied?*
  - Under both partial and total correctness, if  $\sigma \not\models p$ , then the triple is satisfied but we make no guarantees about whether  $M(S, \sigma)$  exists or not or if it satisfies  $q$  or not.

- *What if  $S$  terminates and satisfies the postcondition  $q$ ? Does  $p$  matter?*
  - Under both partial and total correctness, if  $M(S, \sigma)$  exists and  $\models q$ , then the triple is satisfied whether  $\sigma \models p$  or  $\sigma \not\models p$ .
- *What if  $S$  diverges (doesn't terminate)?*
  - Under partial correctness, if  $M(S, \sigma)$  doesn't exist (i.e., the loop diverges) then the triple is satisfied whether  $\sigma \models p$  or  $\sigma \not\models p$ .
  - Under total correctness, if  $M(S, \sigma)$  doesn't exist, then if  $\sigma \models p$ , the triple is not satisfied. (If  $\sigma \not\models p$  then the triple is satisfied regardless.)
- *Trivial partial correctness triples*
  - From the discussion above, we can identify three cases where  $\sigma \models \{p\} S \{q\}$  kind of trivially: (1) When  $p$  is a contradiction: (i.e.,  $\models \neg p$ ), (2) When  $S$  never terminates (so  $M(S, \sigma)$  is never defined), and (3) When  $q$  is a tautology (If  $\models q$ , i.e., if  $q$  is satisfied in all states, then it's certainly satisfied whenever  $M(S, \sigma)$  exists).