

Reasoning About Code; Hoare Logic

Announcements

- Homework 0 due on Friday
 - Questions?
- Office hours schedule finalized at:
 - www.cs.rpi.edu/~milanova/csci2600

Fall 15 CSCI 2600, A Milanova

2

Outline

- Hoare logic
 - Hoare Triples
 - Rules for backward reasoning
 - Assignment, sequence, if-then-else, method calls
- Lab 1 (optional lab: 2pt extra credit on HW1)
 - The Dafny programming language
 - Submit your Dafny program in a *.txt file by midnight tomorrow

Fall 15 CSCI 2600, A Milanova

3

Hoare Logic

- Formal framework for reasoning about code
- Sir Anthony Hoare (Sir Tony Hoare or Sir C.A.R. Hoare)
 - Hoare logic
 - Quicksort algorithm
 - Other contributions to programming languages
 - Turing Award in 1980

Fall 15 CSCI 2600, A Milanova

4

Hoare Triples

- A Hoare Triple: $\{ P \} \text{code} \{ Q \}$
 - P and Q are logical statements about program values, and `code` is program code (in our case, Java code)
- “ $\{ P \} \text{code} \{ Q \}$ ” means “if P is true and we execute `code`, then Q is true afterward”
 - “ $\{ P \} \text{code} \{ Q \}$ ” is a logical formula, just like “ $0 \leq \text{index}$ ”

Fall 15 CSCI 2600, A Milanova (based on slide by Michael Ernst)

5

Examples of Hoare Triples

$\{ x > 0 \} \text{ x++ } \{ x > 1 \}$ is true
 $\{ x > 0 \} \text{ x++ } \{ x > -1 \}$ is true
 $\{ x \geq 0 \} \text{ x++ } \{ x > 1 \}$ is false. Why?

$\{ x > 0 \} \text{ x++ } \{ x > 0 \}$ is ??
 $\{ x < 0 \} \text{ x++ } \{ x < 0 \}$ is ??
 $\{ x = a \} \text{ if } (x < 0) \text{ x} = -x \{ x = |a| \}$ is ??
 $\{ x = y \} \text{ x} = x + 3 \{ x = y \}$ is ??

Fall 15 CSCI 2600, A Milanova

6

Rules for Backward Reasoning: Assignment

```
// precondition: ??
x = expression
// postcondition: Q
Rule: precondition is Q with all occurrences of
x in Q replaced by expression
// precondition: y+1 > 0    // precondition: z+1 > 0
x = y+1;                    z = z+1;
// postcondition: x>0      // postcondition: z>0
```

Fall 15 CSCI 2600, A Milanova (based on slide by Michael Ernst)

7

Weakest Precondition

Rule derives the **weakest precondition**

```
// precondition: y+1 > 0 (equiv. y > -1)
x = y+1
// postcondition: x>0
(y+1)>0 is the weakest precondition for code
x=y+1 and postcondition x>0
Notation: wp stands for weakest precondition
wp("x=expression",Q) = Q with all
occurrences of x replaced by expression
```

8

Rule for Assignment

```
{ wp("x=expression",Q) }
x = expression;
{ Q }
```

Rule: the **weakest precondition**
wp("x=expression",Q) is Q with all
occurrences of x in Q replaced by
expression

9

Aside: Weaker and Stronger Conditions

- "P is stronger than Q" means "P implies Q"
- "P is stronger than Q" means "P guarantees more than Q"
- E.g., $y > 0$ is stronger than $y > -1$

Which one is stronger?

$x > 0 \ \&\& \ y = 0$ or $x > 0 \ \&\& \ y \geq 0$

$0 \leq x \leq 10$ or $0 \leq x \leq 1$

$x = 5 \ \&\& \ y \% 4 = 2$ or $x = 5 \ \&\& \ y$ is even

10

Aside: Weaker and Stronger Conditions

Let the following be true:

$P \Rightarrow Q \quad Q \Rightarrow R \quad S \Rightarrow T \quad T \Rightarrow U$
{ Q } code { T }

Which of the following are true?

- (1) { P } code { T }
- (2) { R } code { T }
- (3) { Q } code { S }
- (4) { Q } code { U }

Fall 15 CSCI 2600, A Milanova (based on slide by Michael Ernst)

11

Aside: Weaker and Stronger Conditions

- In **backward reasoning**, we determine the precondition, given code and a postcondition Q
 - We want the **weakest precondition**, wp(code,Q)
- In **forward reasoning**, we determine the postcondition, given code and a precondition P
 - Normally we want the **strongest postcondition**

Fall 15 CSCI 2600, A Milanova

12

Aside: Weaker and Stronger Conditions

Goal: Prove that $\{P\}$ code $\{Q\}$ is a valid triple

- Forward reasoning:
- Backward reasoning

$\{P\}$	$\{P\}$
code	derive $\{P'\}$
derive $\{Q'\}$	code
$\{Q\}$	$\{Q\}$
Then show $Q' \Rightarrow Q$	Then show $P \Rightarrow P'$

Fall 15 CSCI 2600, A Milanova

13

Rules for Backward Reasoning: Sequence

```
// precondition: ??
s1; // statement
s2; // another statement
// postcondition: Q
Work backwards:
precondition is wp("s1;s2;", Q) = wp("s1;", wp("s2;", Q))
Example:
// precondition: ??
x = 0; // postcondition for x=0; same as
y = x+1; // precondition for y=x+1;
// postcondition: y>0 y = x+1;
// postcondition y>0
```

Fall 15 CSCI 2600, A Milanova (based on slide by Michael Ernst)

14

Rule for Sequence

// find weakest precondition for sequence $s1; s2$ and Q

```
{ wp(s1, wp(s2, Q)) }
s1; // statement
{ wp(s2, Q) }
s2; // another statement
{ Q }
```

Working backwards:

precondition is $wp(s1; s2, Q) = wp(s1, wp(s2, Q))$

15

Exercise

```
{ x + 1 + y > 1 } equiv. to { x + y > 0 }
x = x + 1;
{ x + y > 1 }
y = x + y;
{ y > 1 }
```

Spring 15 CSCI 2600, A Milanova

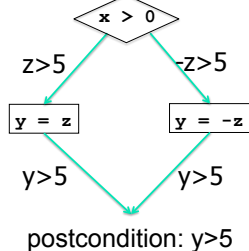
16

If-then-else Statement, Example

// precondition: $?? (z > 5 \ \&\& \ x > 0) \ || \ (z < -5 \ \&\& \ x \leq 0)$

```
if (x > 0) {
    y = z;
}
else {
    y = -z;
}
```

// postcondition: $y > 5$



Fall 15 CSCI 2600, A Milanova

17

Rules for Backward Reasoning: If-then-else

```
// precondition: ??
if (b) s1 else s2
// postcondition: Q
Case analysis, just as we did in the example:
wp("if (b) s1 else s2", Q)
= ( (b && wp("s1", Q))
  || (¬b && wp("s2", Q)) )
```

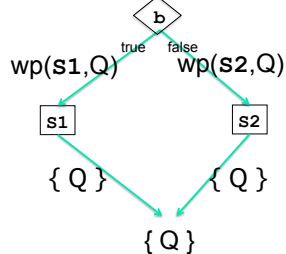
Fall 15 CSCI 2600, A Milanova

18

Rule for If-then-else

// wp: ?? $(b \ \&\& \text{wp}(s1, Q)) \vee (\neg b \ \&\& \text{wp}(s2, Q))$

```
if (b) {
  s1;
}
else {
  s2;
}
{ Q }
```



Fall 15 CSCI 2600, A Milanova

19

Exercise

Precondition: ??

```
z = 0;
if (x != 0) {
  z = x;
} else {
  z = z+1;
}
```

Postcondition: $z > 0$;

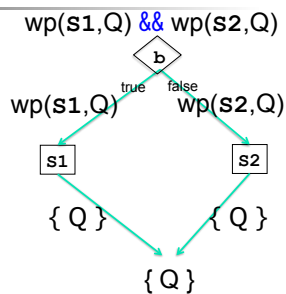
Fall 15 CSCI 2600, A Milanova (example due to Michael Ernst)

20

Why Not This Rule?

// wp: ??

```
if (b) {
  s1;
}
else {
  s2;
}
{ Q }
```



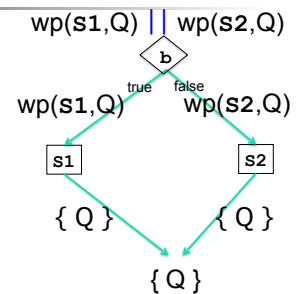
Fall 15 CSCI 2600, A Milanova

21

Why Not This Rule?

// wp: ??

```
if (b) {
  s1;
}
else {
  s2;
}
{ Q }
```



Fall 15 CSCI 2600, A Milanova

22

Exercise

// precondition: ??

```
y = x + 4;
if (x > 0) {
  y = x*x - 1;
}
else {
  y = y+x;
}
```

Postcondition: $\{ y = 0 \}$

$x = x/y$;

// What inputs cause Divide-by-zero?

Find what input causes divide-by-zero at the last statement.

Answer:
Precondition: $x=1 \vee x=-2$

These are the inputs that cause divide-by-zero error

23

If-then-else Statement Review

Forward reasoning Backward reasoning

{ P }	{ $(b \ \&\& \text{wp}("s1", Q)) \vee (\neg b \ \&\& \text{wp}("s2", Q))$ }
if (b)	if (b)
{ P && b }	{ wp("s1", Q) }
s1	s1
{ Q1 }	{ Q }
else	else
{ P && $\neg b$ }	{ wp("s2", Q) }
s2	s2
{ Q2 }	{ Q }
{ Q1 Q2 }	{ Q }

Fall 15 CSCI 2600, A Milanova (based on slide by Michael Ernst)

24

If-then Statement

```
// precondition: ??  
if (x > y) {  
    z = x;  
    x = y;  
    y = z;  
}  
// postcondition: x < y
```

Fall 15 CSCI 2600, A Milanova

25

Rules for Backward Reasoning: Method Call

```
// precondition: ??  
x = foo()  
// postcondition: Q  
If method has no side-effects, just like  
assignment  
// precondition: ??  
x = Math.abs(y)  
// postcondition: x = 1  
Precondition is  $y = 1 \parallel y = -1$ 
```

Fall 15 CSCI 2600, A Milanova (based on slide by Mike Ernst, UW)

26

Summary So Far

- Intro to reasoning about code. Concepts
 - Specifications, preconditions and postconditions, forward and backward reasoning
- Hoare triples
- Rules for backward reasoning
 - Rule for assignment
 - Rule for sequence of statements
 - Rule for if-then-else
 - Method call

Fall 15 CSCI 2600, A Milanova

27

Lab 1

- Dafny: programming language and verifier
 - Author: K. Rustan M. Leino, Microsoft Research
 - Programmer writes programs with specifications
 - Verifier proves that program obeys specification
- Install (on Windows) from <http://dafny.codeplex.com>
- Try online at <http://www.rise4fun.com/dafny>

Fall 15 CSCI 2600, A Milanova

28

Dafny Basics

- The smallest unit of verification is the **method**
method Foo(x: int, y: int) **returns** (z: int, w: int)
- **Preconditions**
requires $x == 0 \ \&\& \ y \geq 0$
- **Postconditions**
ensures $z \neq 0 \parallel w \neq 0$

Fall 15 CSCI 2600, A Milanova

29

Our Earlier Exercise, in Dafny

```
// precondition: ??  
y = x + 4  
if (x > 0) {  
    y = x*x - 1;  
}  
else {  
    y = y+x;  
}  
{ y = 0 }  
x = x/y;
```

Find what input causes
divide-by-zero at the last
statement.

Answer:
Precondition: $x=1 \parallel x=-2$

These are the inputs that
cause divide-by-zero error

30

Our Division Example in Dafny

```
method DivisionByZero(x: int) returns (y: int)
  requires x == 1 || x == -2
  ensures y == 0
{
  y := x + 4;
  if (x > 0) {
    y := x*x - 1;
  } else {
    y := y + x;
  }
}
```

Named returns, can have multiple output variables!

Equality test: ==, NOT =

Assignment: :=, NOT =

31

Lab Exercise

// weakest precondition: ??

```
if (x < 5) {
  y = x*x;
}
else {
  y = x+1;
}
```

// postcondition: $y \geq 9$

Fall 15 CSCI 2600, A Milanova (example due to Michael Ernst)

1. Compute the weakest precondition
2. Verify your answer with Dafny
3. Cut and paste your program into file DafnyCode.txt, then submit it to the Homework Server, Principles of Software Lab 1, Deadline midnight Wednesday .

32