

A Specification Coverage Metric Using a Dynamic Invariant Detector

Rob McGregor and Will Thomasson

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

1

Outline

- ◆ Problem
- ◆ Proposed Solution
- ◆ Background
- ◆ The Method
- ◆ Example
- ◆ Results
- ◆ Evaluation/Conclusion

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

2

Problem

- ◆ We want a reliable way to determine the coverage of a test suite
- ◆ Traditional methods test code coverage
- ◆ No methods to test specification coverage

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

3

Proposed Solution

- ◆ Metric to measure the specification coverage of the test suite
- ◆ Test suite and specification developed together
- ◆ Uses dynamic invariant detection

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

4

Proposed Solution

- ◆ Iterative process
- ◆ Test cases needed are identified by our method
- ◆ Applied to three Java classes and their test suites

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

5

Background

- ◆ Program Invariants
 - Static versus dynamic detection
- ◆ Test Suites
 - A subset of all possible inputs chosen to demonstrate the behavior of all inputs
 - Exhaustively tested

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

6

Background

- Code Coverage Techniques
 - “Black Box” (functional analysis)
 - “White Box” (structural analysis)
- Daikon
 - Dynamic invariant detection
 - Runs test cases on source code
 - Accuracy of invariants depends on accuracy of test cases

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

7

Our Approach (outline)

- Develop an Initial Test Suite
- Establish and Analyze the Specification
- Add Test Cases
- Cycle and Conclude

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

8

Our Approach

- Develop an Initial Test Suite
 - Create predetermined list of invariants
- Establish and Analyze the Specification
 - Run the invariant detector
 - Examine the invariants
 - Separate into lists

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

9

Our Approach

- Add Test Cases
 - Examine the specification to determine which test cases to add
- Cycle and Conclude
 - Repeat the previous two steps
 - Stop when no incorrect invariants are found
 - Accurate test suite
 - Accurate specification

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

10

Date Class

- Overview
 - Data members
 - January 1, 0000
 - monthDays []
- Develop an Initial Test Suite
 - Random dates

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

11

Date Class

- Establish and Analyze the Specification
 - Function: `dateToDays (Date d)`
 - Correct invariant:
 - `d.day <= d.monthDays [d.month-1]`
 - Incorrect invariant:
 - `d.year > d.monthDays [d.month-1]`
 - Counterexample:
 - `d.year <= 28`
- Add Test Cases
 - `Date d = new Date (1,1,0);`
 - `dateToDays (d);`

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

12

Date Class

❖ Cycle and Conclude

- After test cases were added, we ran Daikon on the improved test suite
- If incorrect invariants appeared, we added test cases again
- When no incorrect invariants appeared, we stopped
- Entire process was four iterations

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

13

Results: Matrix

Function Name	First Run	
	Incorrect	Missing
matrix()	0	0
matrix(int)	0	0
isEmpty()	0	0
clear()	0	0
size()	0	2
set(int, int, int)	7	1
setRow(int, int)	6	1
setCol(int, int)	6	1
element(int, int)	5	2
fillWith(int)	0	1
negate()	0	0
SubMatrix(int, int)	0	0
Add(matrix)	0	1
Subtract(matrix)	0	1
Multiply(matrix)	0	0

❖ Simplest Class

- 2-D array elements cannot be analyzed

❖ Only one iteration

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

14

Results: Date

Function Name	First Run			Second Run			Third Run		
	Incorrect	Missing	Tests Added	Incorrect	Missing	Tests Added	Incorrect	Missing	Tests Added
Add(Date)	31	5	3	25	5	3	0	4	0
Date(int, int, int)	3	4	N/A	1	3	N/A	0	3	N/A
Subtract(Date)	47	6	3	36	6	8	0	6	0
Subtract(int)	28	7	4	13	6	3	0	6	0
dateToDays(Date)	7	4	1	5	2	2	0	1	0
daysToDate(int)	6	5	2	1	5	1	0	3	0
equals(Date)	5	13	1	7	13	4	4	12	4
getYear()	6	4	1	6	4	2	0	4	0
isLeapYear(int)	0	4	0	0	4	0	0	4	0
midAnniversary(Date)	17	0	2	12	0	4	1	0	1
toString()	4	0	1	4	0	2	0	0	0

- ❖ Slow convergence from first to second run
- ❖ Fewer tests added than incorrect invariants

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

15

Results: Polynomial

Function Name	First Run			Second Run			Third Run		
	Incorrect	Missing	Tests Added	Incorrect	Missing	Tests Added	Incorrect	Missing	Tests Added
Polynomial()	0	0	0	0	0	0	0	0	0
Polynomial(int [])	7	0	2	0	0	0	0	0	0
Polynomial(int)	4	0	0	0	0	0	0	0	0
zeroPolynomial()	2	0	0	1	0	0	0	0	0
addPolynomial()	38	0	8	6	0	3	0	0	0
multiplyPolynomial()	39	1	8	4	1	4	0	1	0
evalAt(int)	1	0	1	0	0	0	0	0	0
derivative()	3	0	4	0	0	0	0	0	0
raiseTo(int)	24	0	12	12	0	16	2	0	2

- ❖ Quicker convergence of incorrect invariants
- ❖ Complicated invariants, difficult to analyze

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

16

Summary/Evaluation (1/2)

- ❖ Showed feasibility
 - Useful invariants were found
 - Eliminated all incorrect invariants
- ❖ Our initial test suites
 - Random and incomplete
 - Good for our experiment
- ❖ Don't know how our metric compares
 - Is it as good as or better than branch coverage?

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

17

Summary/Evaluation (2/2)

- ❖ Are missing invariants important?
 - Do not imply lack of test coverage
 - Attempting to detect them adds pointless tests
- ❖ Method evolved during experiment
 - We improved our method of adding tests
 - Numbers for our results changed slightly
- ❖ Limitations
 - Invariant detector
 - Scalability

12/15/2003

A Specification Coverage Metric Using
a Dynamic Invariant Detector

18