



Measuring and Mitigating Gaps in Structural Testing

Soneya Binta Hossain

Matthew Dwyer


Sebastian Elbaum

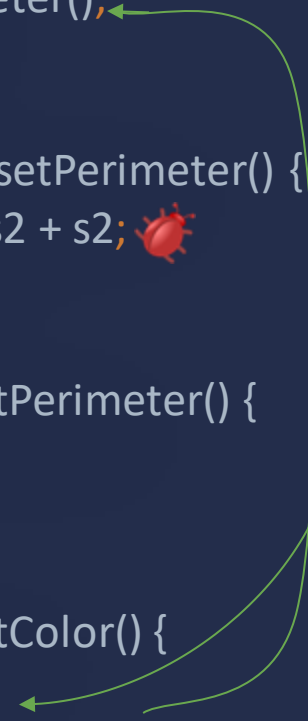
Anh Nguyen-Tuong



Measuring Gaps

	E	E+C	G
1:	✓		✓
2:	✓		✓
3:	✓		✓
4:	✓	✓	
5:	✓		✓
6:	✓		✓
7:	✓		✓
8:	✓	✓	

```
public class Triangle {  
  
    int s1, s2, s3, P, C;  
    Triangle(int a1, int a2, int a3, int c) {  
1:         s1 = a1;  
2:         s2 = a2;  
3:         s3 = a3;  
4:         C = c;  
5:         setPerimeter();  
        }  
  
        private void setPerimeter() {  
6:            P = s1 + s2 + s2;   
        }  
  
        public int getPerimeter() {  
7:            return P;  
        }  
  
        public int getColor() {  
8:            return C;  
        }  
    }  
}
```



```
@Test  
public void testColor() {  
    Triangle t = new Triangle(2,3,2,1);  
    t.getPerimeter();  
    assertEquals(1, t.getColor());  
}
```

Covered: 100%
Checked: 25%
In Gap: 75%

Mitigating Gaps

```
public class Triangle {  
  
    int s1, s2, s3, P, C;  
    Triangle(int a1, int a2, int a3, int c) {  
1:      s1 = a1;  
2:      s2 = a2;  
3:      s3 = a3;  
4:      C = c;  
5:      setPerimeter();  
    }  
  
    private void setPerimeter() {  
6:      P = s1 + s2 + s3;  
    }  
  
    public int getPerimeter() {  
7:      return P;  
    }  
  
    public int getColor() {  
8:      return C;  
    }  
}
```

field write: s1, s2, s3

field read: s1, s2, s3
write: P

field read: P

Recommendation

getPerimeter()

```
@Test  
public void testColor() {  
    Triangle t = new Triangle(2,3,2,1);  
    assertEquals(1, t.getColor());  
    assertEquals(7, t.getPerimeter());  
}
```

Evaluation:

Artifacts

- 13 Java Applications
- 16K tests
- 51.6K test assertions

HOST COVERAGE, HCC AND COVERAGE GAP FOR STATEMENT AND OBJECT BRANCH CRITERIA

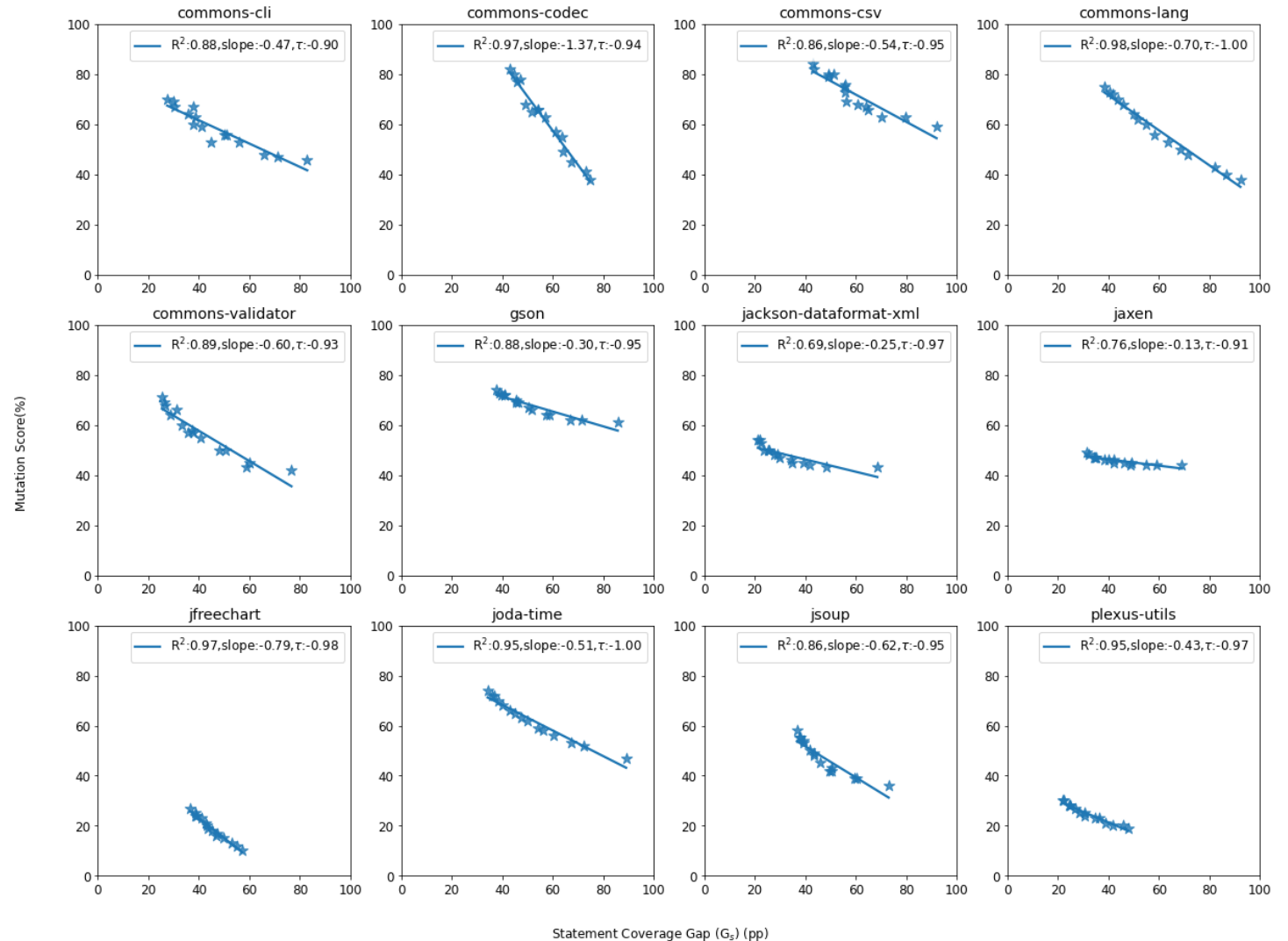
Artifact	Test(#)	Assertion(#)	SC(%)	SCC(%)	Gap _s (pp)	OBC(%)	OBCC(%)	Gap _{ob} (pp)
Commons-Cli	137	405	83	55	28	74	44	30
Commons-Codec	563	1,030	75	32	43	77	32	45
Commons-Csv	278	898	92	49	43	88	41	47
Commons-Lang	2,534	14,153	82	54	28	81	52	29
Commons-Validator	442	2,276	77	51	26	76	46	30
Gson	1,014	1,723	86	48	38	79	46	33
Jackson-Dataformat-Xml	185	530	68	47	21	60	41	19
Jaxen	581	567	67	38	29	56	25	31
JFreeChart	2,174	5,420	57	21	36	47	17	30
Joda-Time	4,193	17,589	89	55	34	77	41	36
Jsoup	510	1,645	73	36	37	73	35	38
Plexus-Utils	277	780	48	26	22	37	18	19
XStream	1,697	1,238	74	25	49	72	21	51
Total/Average:	14.6K	48.3K	75	41	34	69	35	34

Finding: Gaps range from 19-51 percentage points (pp), with an average of 35pp

Impact of Gaps on Fault Detection

Findings:

- There exists a strong negative and statistically-significant correlation between gap size and fault-detection effectiveness.



**PERCENTAGE OF ASSERTION FOCUS METHODS RECOMMENDED
WITHIN THE TOP-K RECOMMENDATIONS ACROSS ALL 13 ARTIFACTS**

Artifacts	Assert(#)	Top 1(%)	Top 5(%)	Top 10(%)
Commons-Cli	332	16	51	70
Commons-Codec	532	84	96	97
Commons-Csv	602	69	84	90
Commons-Lang	9843	80	96	98
Commons-Validator	1441	50	77	89
Jackson-Dataformat-Xml	83	33	43	63
Jaxen	134	11	30	37
JFreeChart	3240	82	93	97
Joda-Time	15775	17	43	53
Jsoup	1098	21	31	38
Gson	871	53	82	87
Plexus-Utils	365	55	75	78
XStream	578	38	56	59
Summary	Total 34894	Average 46	Average 67	Average 73

Recommender Performance

Findings:

- On average, 67% focus methods are within top-5 recommendations
- Nearly 50% of the developer-written focus methods are within top-1 recommendation
- Fault detection improved as much as 57pp and on avg. 13pp