Software Engineering
Università della Svizzera Italiana
Faculty of Informatics
January 6, 2020

Assignment 11: Test Automation

Giorgia Adorni (giorgia.adorni@usi.ch)

1 Introduction

The main purpose of this assignment is to analyse the classes `DirectedAcyclicGraph` and `EdmondsBlossomShrinking` provided by jgrapht resources, using **SonarLint**, an IDE extension that helps you detect and fix quality issues as you write code. Then, **Randoop**, that is a command-line tool used for automatic unit test generation for a given Java program, is used to generate tests with a time budget of 3 seconds.

2 DirectedAcyclicGraph

SonarLint

Running the analysis on `DirectedAcyclicGraph` class, **SonarLint** found the issues shown in Figure 1.

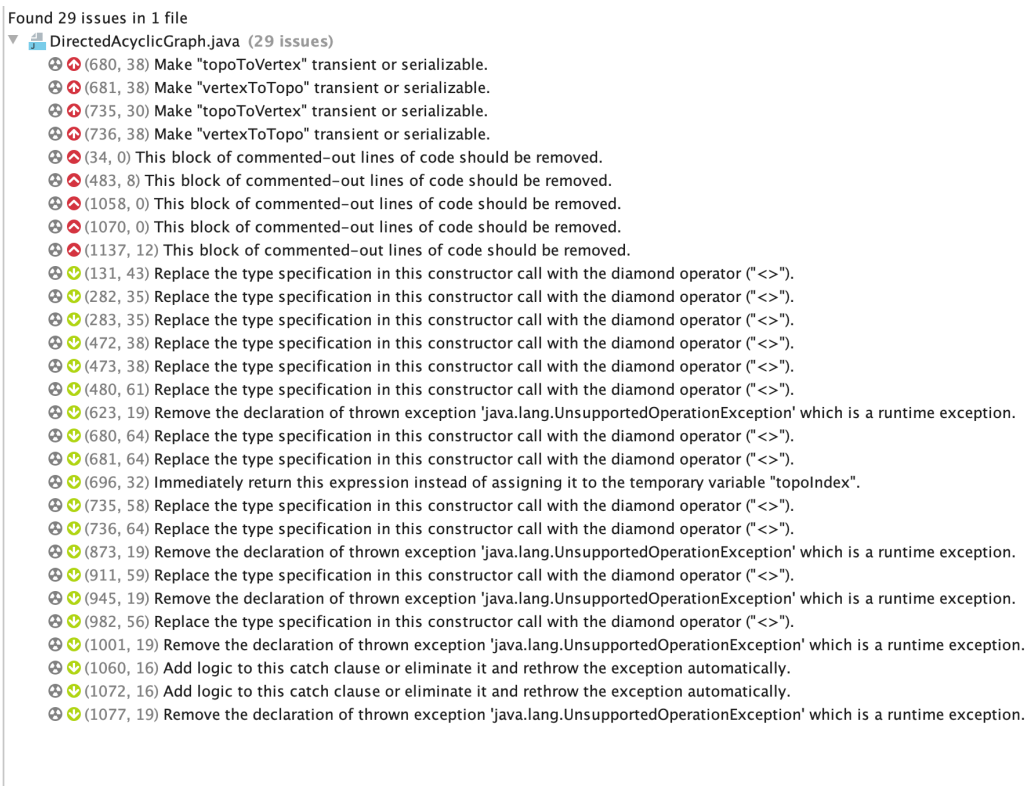

Figure 1: SonarLint issues founded in `DirectedAcyclicGraph`

In total 29 issues are found, in particular, 4 Critical Code Smells, 5 Major Code Smells and 20 Minor Code Smells.

Analysing the severity scale of **SonarLint** issues, the most serious problems identified, defined as Critical Code Smells, are those reporting "Make ... transient or serializable". I completely agree with the seriousness associated with these as they violate the contract of the serializable interface causing exceptions at runtime. All the minor problems identified are all very useful suggestions to the programmer and they can be followed or not as they do not affect the functioning of the program.

Randoop

Randoop is run on `DirectedAcyclicGraph` class with the following command:

```
java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources
randoop.main.Main gentests --classlist=classlist.txt --time-limit=3
```

Listing 1: Randoop Execution

Then, **Randoop** generates an error-revealing test (by default called `ErrorTest0.java`) that is compiled and executed with the following commands:

```
javac -classpath lib/randoop-all-4.2.0.jar ErrorTest0.java -sourcepath src/
```

Listing 2: Compile Error Revealing Test

```
java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources:.
org.junit.runner.JUnitCore ErrorTest0
```

Listing 3: Execute Error Revealing Test

After compiling, 9 different tests are generated.

```
Giorgias-MBP:jgrapht-core-0.9.2-sources giorgia$ java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources randoop.main.Main gentests --classlist=classlist.txt --time-limi
t=3
PUBLIC MEMBERS=90
Explorer = ForwardGenerator(allSequences:0, sideEffectFreeMethods:1117, subsumed_sequences:0, runtimePrimitivesSeen:38)

Progress update: steps=1, test inputs generated=0, failing inputs=0      (Tue Dec 17 09:38:12 CET 2019     9MB used)
Progress update: steps=257, test inputs generated=66, failing inputs=9      (Tue Dec 17 09:38:15 CET 2019     47MB used)
Normal method executions: 3519
Exceptional method executions: 18

Average method execution time (normal termination):      0.00177
Average method execution time (exceptional termination): 0.0744
Approximate memory usage 47MB

Error-revealing test output:
Error-revealing test count: 9
Writing error-revealing JUnit tests...
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/ErrorTest0.java
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/ErrorTest.java
Wrote error-revealing JUnit tests.

About to look for failing assertions in 30 regression sequences.

Regression test output:
Regression test count: 30
Writing regression JUnit tests...
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/RegressionTest0.java
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/RegressionTest.java
Wrote regression JUnit tests.
About to look for flaky methods.

Invalid tests generated: 0
```

Figure 2: Compile Randoop

After the execution, it is possible to see that all the tests thrown `NullPointerException`.

```
Giorgias-MBP:jgrapht-core-0.9.2-sources giorgia$ java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources:. org.junit.runner.JUnitCore ErrorTest0
JUnit version 4.13-beta-3
.E.E.E.E.E.E.E.E.E
Time: 0.008
There were 9 failures:
1) test1(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedArrayImpl.setVisited(DirectedAcyclicGraph.java:1055)
        at ErrorTest0.test1(ErrorTest0.java:16)
2) test2(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.setVisited(DirectedAcyclicGraph.java:864)
        at ErrorTest0.test2(ErrorTest0.java:25)
3) test3(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.setVisited(DirectedAcyclicGraph.java:864)
        at ErrorTest0.test3(ErrorTest0.java:34)
4) test4(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.setVisited(DirectedAcyclicGraph.java:864)
        at ErrorTest0.test4(ErrorTest0.java:43)
5) test5(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedArrayImpl.setVisited(DirectedAcyclicGraph.java:1055)
        at ErrorTest0.test5(ErrorTest0.java:52)
6) test6(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.getVisited(DirectedAcyclicGraph.java:869)
        at ErrorTest0.test6(ErrorTest0.java:61)
7) test7(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedArrayImpl.setVisited(DirectedAcyclicGraph.java:1055)
        at ErrorTest0.test7(ErrorTest0.java:70)
8) test8(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.getVisited(DirectedAcyclicGraph.java:869)
        at ErrorTest0.test8(ErrorTest0.java:79)
9) test9(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.translateIndex(DirectedAcyclicGraph.java:890)
        at org.jgrapht.experimental.dag.DirectedAcyclicGraph$VisitedBitSetImpl.getVisited(DirectedAcyclicGraph.java:869)
        at ErrorTest0.test9(ErrorTest0.java:88)

FAILURES!!!
Tests run: 9,  Failures: 9
```

Figure 3: Randoop execution on `DirectedAcyclicGraph`

```
import org.junit.FixMethodOrder;
import org.junit.Test;
import org.junit.runners.MethodSorters;

@FixMethodOrder(MethodSorters.NAME_ASCENDING)
public class ErrorTest0 {

    public static boolean debug = false;

    @Test
    public void test1() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test1");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedArrayImpl
            visitedArrayImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedArrayImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedArrayImpl0.setVisited((int) (byte) 1);
    }

    @Test
    public void test2() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test2");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedBitSetImpl0.setVisited((int) '4');
    }

    @Test
    public void test3() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test3");
```

```
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedBitSetImpl0.setVisited((int) '#');
    }

    @Test
    public void test4() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test4");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedBitSetImpl0.setVisited((int) (byte) 0);
    }

    @Test
    public void test5() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test5");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedArrayImpl
            visitedArrayImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedArrayImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedArrayImpl0.setVisited((int) (short) 10);
    }

    @Test
    public void test6() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test6");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        boolean boolean2 = visitedBitSetImpl0.getVisited((int) '4');
    }

    @Test
    public void test7() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test7");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedArrayImpl
            visitedArrayImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedArrayImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        visitedArrayImpl0.setVisited((int) (byte) 0);
    }

    @Test
    public void test8() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test8");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
            VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        boolean boolean2 = visitedBitSetImpl0.getVisited((-1));
    }

    @Test
    public void test9() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test9");
        org.jgrapht.experimental.dag.DirectedAcyclicGraph.VisitedBitSetImpl
```

```
            visitedBitSetImpl0 = new org.jgrapht.experimental.dag.DirectedAcyclicGraph.
                VisitedBitSetImpl();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        boolean boolean2 = visitedBitSetImpl0.getVisited(10);
    }
}
```

Listing 4: ErrorTestDirectedAcyclicGraph

All the `NullPointerExceptions` are caused by a similar sequence of incorrect calls. Both the methods `VisitedArrayImpl()` and `VisitedBitSetImpl()` do not define constructors, except the default one that does not do anything. These classes expect firstly a call to the `getInstance(Region region)` method, that **Randoop** does not execute. Hence, all the attributes are set null and this cause the failure of the calling of any other method.

3 EdmondsBlossomShrinking

SonarLint

Running the analysis on `EdmondsBlossomShrinking` class, **SonarLint** found the issues shown in Figure 4.

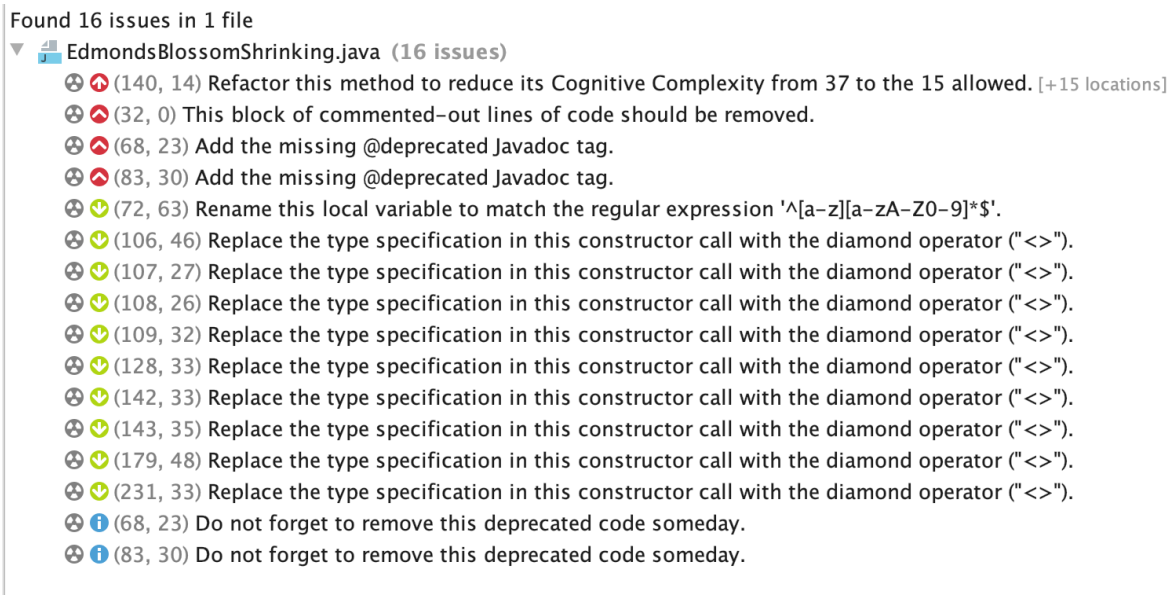

Figure 4: SonarLint issues founded in `EdmondsBlossomShrinking`

In total 16 issues are found, in particular, 1 Critical Code Smells, 3 Major Code Smells, 10 Minor Code Smells and 2 Info Code Smell.

Analysing the severity scale of **SonarLint** issues, contrary to the `DirectedAcyclicGraph` class, in the `EdmondsBlossomShrinking` class, the most serious problems identified report "Refactor this method to reduce its Cognitive Complexity", which curiously I would not have identified as a problem of correctness as I did in the case of the violation of the serializable interfaces. Since the code is highly complex, it is difficult to guarantee its correctness and probably a future modification will lead to errors in the code.

The minor errors are the same as those that appear in the analysis of `DirectedAcyclicGraph`.

Randoop

Randoop is run on `EdmondsBlossomShrinking` class with the following command

```
java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources
randoop.main.Main gentests --classlist=classlist.txt --time-limit=3
```

Listing 5: Randoop Execution

Then, **Randoop** generates an error-revealing test (by default called `ErrorTest0.java`) that is compiled and executed with the following commands:

```
javac -classpath lib/randoop-all-4.2.0.jar ErrorTest0.java -sourcepath src/
```

Listing 6: Compile Error Revealing Test

```
java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources:.
org.junit.runner.JUnitCore ErrorTest0
```

Listing 7: Execute Error Revealing Test

After compiling, only 1 test is generated.

```
Giorgias-MBP:jgrapht-core-0.9.2-sources giorgia$ java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources randoop.main.Main gentests --classlist=classlist.txt --time-limi
t=3
PUBLIC MEMBERS=6
Explorer = ForwardGenerator(allSequences:0, sideEffectFreeMethods:1117, subsumed_sequences:0, runtimePrimitivesSeen:38)

Progress update: steps=1, test inputs generated=0, failing inputs=0      (Tue Dec 17 09:36:40 CET 2019     22MB used)
Progress update: steps=212, test inputs generated=65, failing inputs=1      (Tue Dec 17 09:36:43 CET 2019     70MB used)
Normal method executions: 339
Exceptional method executions: 4

Average method execution time (normal termination):      0.0100
Average method execution time (exceptional termination): 0.227
Approximate memory usage 70MB

Error-revealing test output:
Error-revealing test count: 1
Writing error-revealing JUnit tests...
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/ErrorTest0.java
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/ErrorTest.java
Wrote error-revealing JUnit tests.

About to look for failing assertions in 42 regression sequences.

Regression test output:
Regression test count: 42
Writing regression JUnit tests...
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/RegressionTest0.java
Created file /Users/giorgia/test-automation/code/jgrapht-resources/jgrapht-core-0.9.2-sources/RegressionTest.java
Wrote regression JUnit tests.
About to look for flaky methods.

Invalid tests generated: 0
```

Figure 5: Compile Randoop

After the execution, it is possible to see that also, in this case, the test thrown a `NullPointerException` as for the `DirectedAcyclicGraph` class. In this case, the exception is threw when the instance method of a null object is called.

```
Giorgias-MBP:jgrapht-core-0.9.2-sources giorgia$ java -classpath lib/randoop-all-4.2.0.jar:out/production/jgrapht-core-0.9.2-sources:. org.junit.runner.JUnitCore ErrorTest0
JUnit version 4.13-beta-3
.E
Time: 0.007
There was 1 failure:
1) test1(ErrorTest0)
java.lang.NullPointerException
        at org.jgrapht.alg.EdmondsBlossomShrinking.findMatch(EdmondsBlossomShrinking.java:111)
        at org.jgrapht.alg.EdmondsBlossomShrinking.getMatching(EdmondsBlossomShrinking.java:94)
        at ErrorTest0.test1(ErrorTest0.java:16)

FAILURES!!!
Tests run: 1,  Failures: 1
```

Figure 6: Randoop execution on `EdmondsBlossomShrinking`

```java
import org.junit.FixMethodOrder;
import org.junit.Test;
import org.junit.runners.MethodSorters;

@FixMethodOrder(MethodSorters.NAME_ASCENDING)
public class ErrorTest0 {

    public static boolean debug = false;

    @Test
    public void test1() throws Throwable {
        if (debug)
            System.out.format("%n%s%n", "ErrorTest0.test1");
        org.jgrapht.alg.EdmondsBlossomShrinking<java.lang.Comparable<java.lang.String>,
            java.lang.CharSequence> strComparableEdmondsBlossomShrinking0 = new org.
            jgrapht.alg.EdmondsBlossomShrinking<java.lang.Comparable<java.lang.String>,
            java.lang.CharSequence>();
        // during test generation this statement threw an exception of type java.lang.
            NullPointerException in error
        java.util.Set<java.lang.CharSequence> charSequenceSet1 =
            strComparableEdmondsBlossomShrinking0.getMatching();
    }
}
```

Listing 8: ErrorTestEdmondsBlossomShrinking

The cause of the `NullPointerException` is a sequence of incorrect calls. First of all, the `EdmondsBlossomShrinking` object is created using a deprecated constructor that does not require a graph and does not initialise the graph field. Then, during the call of the method `getMatching()`, the method `vertexSet()` is called on an instance of a null object since the graph is `null`.