A.I. for Software Testing & Reverse Engineering – CS4110

A

E A
A

I

A

Docker Run

F

docker run -it CONTAINER_NAME /bin/bash

Copy the RERS problems in the docker container

B E ***Buggy_RERS_ASTOR.zip***

home/str/

F E H
A E

Build the buggy RERS problem

I A **Problem1_buggy**
I **/home/str**

cd /home/str/

Problem1_buggy
J A

(maven pom file)
E
(contains dependencies to run the test)

(source code of the buggy version of Problem1.java)
(it contains the test caseProblem1.java)
(it contains the binaries)

cd /home/str/Problem1_buggy
mvn clean
mvn install -DskipTests

Run ASTOR

F A F **/home/str/** **astor**
rm -r astor

A
git clone https://github.com/SpoonLabs/astor.git

A
mvn install -DskipTests=true

mvn dependency:build-classpath -B | egrep -v "(^\[INFO\]|^\[WARNING\])" | tee /tmp/astor-classpath.txt

A

B

cd /home/str/astor

java -cp $(cat /tmp/astor-classpath.txt):target/classes fr.inria.main.evolution.AstorMain -mode jgenprog -srcjavafolder /src/main/java/ -srctestfolder /src/test/java/ -binjavafolder /target/classes/ -bintestfolder /target/test-classes/ -location /home/str/Problem1_buggy/ -maxtime 5 -maxgen 3000 -scope local -population 8 -operatorspace relational-Logical-op

A
A **output-astor**

L

A

1) **java -cp $(cat /tmp/astor-classpath.txt):target/classes**
   A
2) **fr.inria.main.evolution.AstorMain**
3) **-mode jgenprog**
   I G J
4) **-srcjavafolder**

l

5) **-srctestfolder**

l

6) **-binjavafolder**

l

7) **-bintestfolder**

l

8) **-location**

l l

Problem1_buggy **/home/str/Problem1_buggy**

9) **-maxtime**

10) **-maxgen 3000**

11) **-scope**

12) **-population 8**

13) **-operatorspace**

F A

L

NB: In this guide, we used only Problem1_buggy as an instance. You can follow the same procedure for the other programs. You would need to update the parameter -location for ASTOR