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

This document provides instructions on how to run ASTOR on the buggy versions of the RERS problems. ASTOR is already installed on the docker container, but you are free to run ASTOR on your local machine if you wish to do so.

The commands that are provided here have been tested on the docker container. We cannot guarantee that the commands will also work on your local machine. If there are any questions on getting ASTOR to run on the docker container, please drop a message on the Mattermost channel.

Docker Run

First, run the docker container using the command:

docker run -it CONTAINER_NAME /bin/bash

Copy the RERS problems in the docker container

On Brightspace we published several buggy RERS problems (see ***Buggy_RERS_ASTOR.zip***). We need to copy these files inside the docker container. You can copy the files into **home/str/**.

For the next steps, we focus on one instance of the RERS problems for simplicity. However, you can follow the same instructions to run ASTOR on the other RERS problems.

Build the buggy RERS problem

In this example, we have run ASTOR on **Problem1_buggy**.
Inside your docker container, go to the folder **/home/str**

Use the command:

cd /home/str/

The folder **Problem1_buggy** folder contains a maven project for the buggy version of Problem1. Java. As maven project, the folder has the following structure:

- /Problem1_buggy
- / Problem1_buggy /pom.xml **(maven pom file)**
- / Problem1_buggy /RERS_buggy.iml
- / Problem1_buggy /lib **(contains dependencies to run the test)**
- / Problem1_buggy /out
- / Problem1_buggy /src/main **(source code of the buggy version of Problem1.java)**
- / Problem1_buggy /src/test **(it contains the test caseProblem1.java)**
- / Problem1_buggy /target **(it contains the binaries)**

Now, let's build the maven project using the commands:

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

Run ASTOR

Finally, it is time to run ASTOR. First navigate to **/home/str/** and remove the existing **astor** folder using **rm -r astor**

Then clone ASTOR using the following command:

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

Once you are done with cloning ASTOR, run the following command:

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

To run ASTOR on the maven project for Problem1 (we previously compiled), use the following steps:

1) Be sure you are in the right folder. Thus, use the command:

cd /home/str/astor

2) run the command:

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

3) Once ASTOR terminates, you should be able to see the result of the search process on the terminal. ASTOR patches are located in the **output-astor** folder. You can find more details about the output of the tool in the following link:
https://github.com/SpoonLabs/astor/blob/master/docs/getting-starting.md

Now, let's break down the parameters for ASTOR:

1) **java -cp $(cat /tmp/astor-classpath.txt):target/classes** This is just the java command to run the binary of ASTOR from the command line with a specific classpath.
2) **fr.inria.main.evolution.AstorMain**
   This is the main class (entry point of the tool)
3) **-mode jgenprog**
   This set up the patch generation engine. In our case, we use GenProg for Java code
4) **-srcjavafolder**

It specifies the folder with the source files for the production code

5) **-srctestfolder**
   It specifies the folder with the source files for the test code
6) **-binjavafolder**
   It specifies the folder with the binaries for the production code
7) **-bintestfolder**
   It specifies the folder with the binaries for the test code
8) **-location**
   It specifies the location of the maven project we are considering. In our case, we consider the problem **Problem1_buggy** in the folder **/home/str/Problem1_buggy**
9) **-maxtime**
   This specifies what is the maximum execution time.
10) **-maxgen 3000**
    This specifies the maximum number of generations to be executed.
11) **-scope**
    This specifies the repair.
12) **-population 8**
    This specifies the population size that should be use.
13) **-operatorspace**
    This specifies that we should look at mutation of operators instead of statements which is set by default.

For other parameter values, you can find more detailed information about on to run ASTOR from the command line at the following website:
https://github.com/SpoonLabs/astor/blob/master/docs/getting-starting.md

1) **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**