CS6004: Code Optimization for Object-Oriented Languages

Spring 2024 (Due: March 5th, 2024)

Assignment PA2: Catch Me If You Can.

1 Assignment Objective

Use Soot to write an intraprocedural flow-sensitive method-escape analysis to find the list of escaping objects for each method over Jimple programs.

2 Detailed Specification

Method-Escape analysis determines whether an object escapes or can be accessed beyond its allocating method. If the object can indeed be accessed outside the allocating method, the escape analysis identifies it as escaping. Your objectives in this assignment is to:

- **Generate intraprocedural points-to information**: You need to generate flow-sensitive intraprocedural points-to information for each method where you can use any data-structure for maintaining the points-to-graph.
- **Perform Escape Analysis**: Using the points-to information generated, perform method-escape-analysis and generate the list of escaping objects (represented using the line number of the allocation site) for each method sorted in lexicographic order.
- For simplicity, you can assume that there will not be any method overloading in the testcases.
- We will provide a “**pa2-starter.zip**“ file that you can expand upon for the assignment.

3 Sample Input Program

```
 1 public class Node {
 2        Node f;
 3        Node g;
 4 }
 5
 6 public class Main {
 7     public static Node global;
 8     public static void main(String[] args) {
 9         foo();
10     }
11     public static Node foo(){
12         Node x = new Node(); // $O_{12}$
13         x.f = new Node(); // $O_{13}$
14         x.f.g = new Node(); // $O_{14}$
15         Node y = new Node(); // $O_{15}$
16         Node z = new Node(); // $O_{16}$
17         y.f = z;
18         bar(x.f, y);
19         return y.f;
20     }
21     public static void bar(Node p1, Node p2){
22         Node w = new Node(); // $O_{22}$
23         w.f = new Node(); // $O_{23}$
24         p2.f = w.f;
25     }
26 }
```

Expected Output:

```
Main:bar 23
Main:foo 13 14 15 16
```

4 Submission

Your submission must be named `rollnum-pa2.zip`, where rollnum is your roll-number in small letters. Upon unzipping the submission, we should get a directory named `rollnum-pa2`. The main class inside this directory should be named `PA2.java`. `PA2.java` should contain all the flags passed to soot along with invoking the escape analysis. Your submission should consist only of Java files (that is, no jars or class files). We will test the code using the version of Soot uploaded on piazza.

For evaluating the assignment, for a testcase Test.java, we would run the following commands:

- `javac -g Test.java`
- `javac -cp .:sootclasses-trunk-jar-with-dependencies.jar PA2.java`
- `java -cp .:sootclasses-trunk-jar-with-dependencies.jar PA2`

If the printed output matches the expected output, you will get the marks for the corresponding testcase.

5 Plagiarism Warning

You are allowed to discuss publicly on piazza, but are supposed to do the assignment completely individually. If plagiarism is found:

- First instance: 0 marks in the assignment
- Second instance: FR grade in the course
- Third instance: report to institutional committee

-*-*-*- Do the assignment honestly, enjoy learning the course. -*-*-*-