CS5863: Assignment 2
Intra-procedural , Flow-sensitive
May-Alias Analysis
Due 21st, March at 11.59 PM

Introduction

In this assignment, you are required to write an analysis pass in LLVM (version 12.0.1) to perform intra-procedural, flow-sensitive may alias analysis based on points-to information for an input program.

Given the LLVM IR of an input program you have to answer the following:

1 The points-to information of all the pointers in the program as observed in the analysis results of the last program point of a function.

2 Yes/No answers w.r.to the alias relationships from every pointer to every other pointer using the points-to information reported (1) above.

Please see the example below for clarity:

```
1  int foo(){
2    int a=10,b=11;
3    int* x=&a;
4    int *y=&b;
5    int *p;
6
7    p=x;
8    // last program point for function foo
9    return a;
10 }
```

Listing 1: Example Source Code

```
1  define dso_local i32 @foo() #0 {
2    %1 = alloca i32, align 4
3    %2 = alloca i32, align 4
4    %3 = alloca i32*, align 8
5    %4 = alloca i32*, align 8
6    %5 = alloca i32*, align 8
7    store i32 10, i32* %1, align 4
8    store i32 11, i32* %2, align 4
9    store i32* %1, i32** %3, align 8
10   store i32* %2, i32** %4, align 8
11   %6 = load i32*, i32** %3, align 8
```

```
12   store i32* %6, i32** %5, align 8
13   %7 = load i32, i32* %1, align 4
14   ret i32 %7
15 }
```

Listing 2: Translated LLVM IR

Output for the above example would be:

Points-to information as observed at the last program point in function `Foo`:

Pointer Var %3={%1}
Pointer Var %4={%2}
Pointer Var %5={%1}
Pointer Var %6={%1}

Alias relationships between the pointers in function `Foo`:

	%3	%4	%5	%6
Pointer Var %3	Yes	No	Yes	Yes
Pointer Var %4	No	Yes	No	No
Pointer Var %5	Yes	No	Yes	Yes
Pointer Var %6	Yes	No	Yes	Yes

Input Format

Input: LLVM IR of input test case on which your analysis pass would run on.

Example:
$ *opt -load $LLVM_BUILD/lib/FlowSensitiveAA.so -fsaa test-ll-file.ll*

Note: Please refer to submission guidelines below for naming conventions and other assumptions.

Output Format

Same as shown in the example above.
Note: Print the points-to information in the ascending order of pointer names

Assumptions

You may make the following assumptions

- The input program would contain only one function other than the main function. (Please refer to the public test cases to get a clear picture.)
- Test cases can contain branches, loops and re-directions too (double pointers, triple pointers, etc.).

Evaluation Criteria

You will be evaluated on both public and private test cases. The assignment files contain the public test cases.

- 20% of the total points of the assignment will be awarded on passing public test cases
- Remaining 80% of the total points of the assignment will be awarded on passing private test cases

Submission guidelines

Please follow the guidelines below to avoid any penalty

- Please submit a zip file named in the format *<ROLLNO_ASSIGN2>* with your code and a working CMakelist
- The name and the flag of the pass should be: *FlowSensitiveAA* and *fsaa* respectively.
- Please adhere to the output format as your code will be evaluated using a script and any spurious character in the output may lead to wrong output.
- Any form of plagiarism will lead to a heavy penalty.