Code analysis

and

transformation

Homework H1

1 Description

Write an LLVM pass to print statistics about invocations of functions included in the CAT API, which is next described.

Specifically, for each bitcode function, you must print

- the name of a CAT function that is invoked in the current bitcode function, and
- the number of instructions of that function that invoke it

The order of CAT functions to print is

1. `CAT_add`
2. `CAT_sub`
3. `CAT_new`
4. `CAT_get`
5. `CAT_set`

Finally, CAT functions that are not invoked by a bitcode function are not printed.

2 CAT sources

You can find the CAT API in CAT.h available in the distributed tests.

3 Example

Consider the following program:

```
#include <CAT.h>

int CAT_execution (void){
  CATData d1;
  CATData d2;
  CATData d3;

  d1  = CAT_new(5);
  d2  = CAT_new(8);
  d3  = CAT_new(0);

  CAT_add(d3, d1, d2);

  return CAT_get(d3);
}

int main (int argc, char *argv[]){
  return CAT_execution();
}
```

your pass must generate the following output (stored in `compiler_output`):

```
H1: "CAT_execution": CAT_add: 1
H1: "CAT_execution": CAT_new: 3
H1: "CAT_execution": CAT_get: 1
```

`H1.tar.bz2` includes a few programs you can use to test your work.

Run all tests Go to `H1/tests` and run `make` to test your work.
The following output means you passed all tests:

```
./misc/run_tests.sh
SUMMARY: 5 tests passed out of 5
```

If you didn't pass a test, then the output will include all tests that have failed.

4 LLVM API and Friends

This section lists the set of LLVM APIs I have used in my H1 solution that I did not use for the past assignment H0. You can choose whether or not using these APIs.

- Method `getFunction` of the class `Module`
- `isa<LLVM CLASS>(LLVM OBJECT)`. For example, `isa<CallInst>(i)` where `i` is an instance of the class `Instruction`
- `cast<LLVM CLASS>(LLVM OBJECT)`. For example, `CallInst *callInst = cast<CallInst>(i)` where `i` is an instance of the class `Instruction`
- `getCalledFunction` of the class `CallInst`
- Method `write_escaped` of the class `raw_ostream`

Next are some headers you might find useful.

```
#include "llvm/Pass.h"
#include "llvm/IR/Module.h"
#include "llvm/IR/Function.h"
#include "llvm/IR/Instructions.h"
#include "llvm/Support/raw_ostream.h"
#include "llvm/Transforms/IPO/PassManagerBuilder.h"
```

5 What to submit

Submit via Canvas the C++ file you've implemented (CatPass.cpp).

For your information: my solution for H1 added 63 lines of C++ code to H0 (computed by `sloccount`).

Good luck with your work!