CS5130 Project 1: Basic Block Histogram–Introduction to LLVM

Due: Wednesday, Sept. 14 @ 5:00 PM

We will be using the Low Level Virtual Machine (LLVM) compiler infrastructure for our project work. LLVM was originally developed at the University of Illinois Urbana-Champaign and has been widely adopted by software industry. In this project you will write a simple program analysis and familiarize yourself with the LLVM source code structure. Your pass will perform a simple basic block analysis to produce histograms showing the distribution of basic block size.

1 Install LLVM

First download, install, and build LLVM/Clang 14.0.6 source code from `http://llvm.org`. Go to the website, click "LLVM 14.0.6" in the "Download!" section. You will go to the `https://github.com/llvm/llvm-project/releases/tag/llvmorg-14.0.6` page. To ensure consistency of my project design, we will use LLVM/Clang 14.0.6 across the whole semester. Do not update from git after you download and build the project. Please download the LLVM source code (llvm-14.0.6.src.tar.xz), CLang source code (clang-14.0.6.src.tar.xz), and LLVM Testsuite (test-suite-14.0.6.src.tar.xz). Untar the three tar balls into a directory of your own choice. It is a good idea to rename llvm-14.0.6.src to llvm and clang-14.0.6.src to clang. Follow the instructions on the website `https://clang.llvm.org/get_started.html` to build LLVM and Clang for your particular machine configuration. For fast compilation, you can use "make -j 4" in the last step. You make need to add `-DLLVM_INCLUDE_BENCHMARKS=OFF` as an cmake option if you see a cmake error as following:

```
CMake Error at CMakeLists.txt:1256 (add_subdirectory):
  add_subdirectory given source
  "llvm/../third-party/benchmark" which is not an existing directory.
```

Peruse through the documentation at `http://llvm.org/docs/`. The LLVM Programmer's Manual (`http://llvm.org/docs/ProgrammersManual.html`) and Writing an LLVM Pass tutorial (`http://llvm.org/docs/WritingAnLLVMPass.html`) are particularly useful.

2 Create a Pass

The `hello world` pass in the LLVM Pass tutorial is located under `$llvm-src/lib/Tansforms/Hello`. You can use this pass as a template. Create a directory `BasicblockHist` within the `$llvm-src/lib/Transforms` directory tree. Copy the files under `Hello` to `BasicBlockHist` and update the file names, structure and class names appropriately.

If you want your pass to build as part of the overall LLVM build you need to register the `BasicBlockHist` directory to the cmake file, `$llvm-src/lib/Transform/CMakeLists.txt`. Your duplicated BasicBlockHist pass implements an LLVM pass that does nothing but print out `Hello` and function names. Before writing any code, make sure you can properly run this dummy pass. Use the attached `test.c` and compile it to LLVM bitcode:

```
clang -O0 -emit-llvm -c test.c
```

Inspect the result using llvm-dis:

```
llvm-dis test.bc
```

This will create a file `test.ll` that should look very similar to the attached test.ll file.

Now try running the dummy BasicBlockHist pass on the bitcode using the `opt` command:

```
opt -enable-new-pm=0 -load $build/lib/LLVMBasicBlockHist.so -basicblockhist
test.bc > /dev/null
```

If all goes well you should see `Hello: erk` print out to `stderr`. Note that $build is the directory you build llvm and I changed the registered pass option from `hello` to `basicblockhist`.

3 Basic Block Analysis Implementation

The next step is to extend `BasicBlockHist.cpp` to perform a simple statistical pass that generates two histograms for basic block distribution in terms of basic block size, which is the number of instructions contained in a basic block. You are also asked to output the average basic block size. The first histogram considers all instructions while the second only count the `load` instructions. The resulting output for `test.bc` should be like the following:

```
Function: erk

All instructions
4: 3
10: 1
12: 1
avg: 6.80

Load only
1: 2
2: 2
4: 1
avg: 2.00
```

The output tells that for function `erk` there are three blocks with size 4 and one with size 10, and one with 12 when all instructions are included. When only the load instructions are counted, the size of basic block `for.cond` is now 2 as there are two `load` instructions in the beginning.

You are asked to output two histograms, for each function, led by the function name. In addition, output two summary histograms at the end which summarize the overall basic block distribution.

4 Test and Submission

You can test your code using the Stanford benchmark suite under test-suite/SingleSource/Benchmarks/Stanford. You should submit, through Canvas,

- your copy of `BasicBlockHist.cpp` (All of your source code should be contained in this file. You should not modify any other files in the LLVM source tree. Please remove any debug output from your code.)
- the basic block histogram for each benchmark in the Stanford suite when using Clang option "-O0" to generate bitcode (please use a file name in the style of banchmarkname.o0.txt)
- the basic block histogram for each benchmark in the Stanford suite when using Clang option "-O3" to generate bitcode (please use a file name in the style of banchmarkname.o3.txt)
- a short report that describes your code design and the impact of the Clang options on basic block size distribution.