

Homework #5

cpe 631

Kyle Ray

February 26, 2018

Contents

[Answers 1](#_Toc491808573)

[Output 1](#_Toc491808574)

[Source Code 1](#_Toc491808575)

[Appendix 1](#_Toc491808576)

# Purpose

To become familiar with the Intel PIN and learn more about measuring the performance of modern computer systems with specific benchmarks.

# Problem 1

The inscount0.cpp pin tool will count the total number of instructions executed by the application it is monitoring and write the result to an output file. In this situation the application being monitored is running a SPEC benchmark. The inscount0 PIN tool was ran against the 621\_wrf\_s and 623\_xalanchbmk\_s benchmarks both with the train input set. The data for the experiment can be found in the table below.

**Table 1. Instruction Count for SPEC Benchmarks via Intel PIN Tool**

|  |  |
| --- | --- |
| Benchmark | Total Instructions Executed |
| 621\_wrf\_s train | 583459817340.00 |
| 623\_xalanchbmk\_s train | 257156226730.00 |

# Problem 2

Modify the inscount0.cpp to count the number of basic blocks, number of memory reads, number of memory writes, and the total number of executed instructions. Profile single-threaded applications and output the results to a file. Demonstrate the application on simple benchmarks matrix\_multiplication and accumulating elements of an array as well as profile at least two benchmarks 621.wrf\_s and 657.xz\_s. Note: simple benchmarks can be found at /apps/arch/arch.tut/simpleBenchmarks

# Bonus

Count and print the statistics for each thread in a separate column for a multithreaded program. This will require modifying the pin tool from problem 2 and making it work with multi-threaded applications. Note: that the pin tool will not generally work with pthreads or other multithreaded paradigms, there is a pin tool thread api that is provided and the code written for this problem should make use of that.

# Problem 3

Design and implement a PIN instrumentation tool for profiling dynamic basic blocks (or streams) in a program. A dynamic basic block is defined as a sequential run of instructions that starts with an instruction that is a target of a taken branch and ends with the first taken branch in a sequence.

# Appendix

Command to build and then run inscount0

Make: make obj-intel64/inscount0.so

Instruction API reference for PIN can be found here:

# <https://software.intel.com/sites/landingpage/pintool/docs/81205/Pin/html/group__INS__BASIC__API.html>