COMP25212 – Laboratory Exercise 4
Measuring Computer Performance

Duration: 1 x two-hour lab session.
Resources: Any computer with a C compiler. This script is intended for those using Linux, but if you wish to use Windows, (or Solaris, FreeBSD or RiscOS) you are welcome to do so. But beware – some of the steps of the process may be different, and you'll need to look for a workaround yourself.

AIMS

To investigate, by running a series of tests, the different subsystems of the computer, including storage, memory and processor.

LEARNING OUTCOMES

- To understand how to use a publicly available set of microbenchmarks, and to understand they may have limitations.
- To understand the behaviour of the different devices that for a computer.
- To learn how to plot and interpret results.

INTRODUCTION

In this lab session, we will continue to measure the performance of the PC you are using – but this time we will use **lmbench**, a suite of programs freely available under GPL. We will be measuring aspects of the file system and disk performance, in connection with the lecture material on disk subsystems. Remember that you must use "submit" in the usual way to show that you completed your work by the deadline. All the labwork MUST be marked in the following lab session unless there is a good excuse or it will be zeroed. This is to avoid tedious marking sessions at the end of the course as well as for you to remember the work you did, which is needed for marking as the TAs are instructed to go in detail over your work.

PREPARATION

The file containing the benchmark (lmbench-3.0-a9.tar) is located in the lab repository, in the "ex4" sub-directory, along with the answer template file (answer.txt). Remember to write your response to the questions below in answer.txt , as this is the file you need to commit, tag and push to Gitlab in order to get marked.

Extract the individual files using the command:

```
$ tar xvf ./lmbench-3.0-a9.tar
```

You will see that this creates files in a sub directory lmbench-3.0-a9 . For the rest of this lab, we will be concerned with the src , bin and doc sub-sub-directories.

First, you will need to compile the benchmarks:

```
$ cd src
$ make lmbench
```

This will (should?) compile all of the individual microbenchmarks into binaries in the bin/i686-pc-linux-gnu directory.

Reflection: you will see that there are a number of compilation warnings – do you expect these in software that is publicly available?

Either modify your `PATH` variable to include this directory in your search path, or remember to execute individual programs from this directory.

Manual pages for the individual benchmarks are available in the "doc" directory. You can display them with:

```
$ man -l ./mhz.8
```

or use `-t` to generate printable PostScript. Remember this if you need documentation on individual programs command lines!

Note – to get repeatable benchmark results on a computer, you should have nothing else running. It is worth rebooting your machine at the beginning of the lab session, as other people may leave software running if they have not logged out cleanly.

PART 1 – (3 mark)

Run the program `mhz` Record the final line of output here: _____

What value CPU MHZ is reported in `/proc/cpuinfo` ? _____

Why are (might be) the values different? _____

_____

PART 2 – (4 mark)

Use the program `lat_mem_rd` to repeat the plot of memory access time against working set size you made in Exercise 1, preferably plotting it together with a different pen colour. (Show this plot to your demonstrator) Are the results comparable? _____

What might explain slight differences between the results?

_____

_____

PART 3 – (4 marks)

WARNING – do not use the programs which test file or disk performance on an NFS-mounted filestore (like your home directory!) You will make the whole school of Computer Science very unhappy as you saturate the file server **WARNING** – use these programs only to read and write to a local hard disk – use:

```
$ df /tmp /var/tmp
```

to find a directory that is mounted locally, i.e. on either:

```
/dev/sda1
/dev/mapper/VolGroup....
```

Use the program `lmdd` to create an 8 megabyte file in (eg) `/var/tmp` :

```
$ lmdd if=internal of=/var/tmp/your-name count=1k
```

What write bandwidth does lmdd report? _____

How is this possible? _____

Note: the maximum physical data transfer rate available from a disk drive connected by a SATA interface is about 3 gigabits per second.
Hint: look at Part 5

PART 4 – (5 marks)

Try creating files of different sizes (up to hundreds of megabytes, if you have the disk space) and plot the effective disk bandwidth. What disk bandwidth is reported for reading files of these different sizes?

Hint: the program `vmstat` can be used to report input/output transfers.

PART 5 – (4 marks)

The `stream` benchmark reports the memory bandwidth available for simply copying memory (and lots of other things). Why do parts 3 and 4 *not* achieve this rate of transfer for small files?

[OPTIONAL PART 6 – (5 marks)]

Use the program `cache` to investigate the cache hierarchy of your machine. It may take several minutes to run. What does it report?

Explain these results

[OPTIONAL PART 7 – (4 marks)]

Investigate the behaviour of random reads from a large file and random writes to a large file and plot the results.