**Project 4 Report – Thi Quynh Ha Nguyen**

This document is the report the Project 4 for CS3013 at WPI for the comparison between the performance of standard file I/O using the read() system call for input with memory-mapped I/O where the mmap() system call is used. The program for this project is called proj4 which takes in a file name as the command-line argument and compute the percentage of printable characters in file. In this project, both printable character as well as space, newline,… are consider “printable”. The above data are the result of running ./proj4 program under five configurations for input files of different sizes 1, 1K, 4K and 8K bytes as well as memory mapped I/O. In the figure 1, the X axis indicates the file size in bytes of different size file in data while the Y axis shows the clock-time of the program while using buffer and memory maps. Overall in the chart, as the file size increase, the amount of elapse time for the command to execute also raise. After the file reached to beyond 20K bytes file size, the response time for call raise can be seen clearly as the linear line on the chart. Except for the buffer size is 1 byte, the differences between different buffer ( 1K to 8K) are not significant. In contrast, when mmap() is used in system call, the time clock difference between each file size are minimal to none and only raised as a significant large file is introduced to system. In comparison from the data from the chart, the performance of memory map is similar to the performance of the buffer with the size of 1024 (the default buffer size). If from the data from the chart, the memory map I/O performed better and more stable while read system call’s result values might value if different size of buffer is used. However, this might not indicates the overall performance of each system calls as more tests are needed to prove whether one is significant better than the others.