UNIVERSITY OF PRETORIA, DEPARTMENT OF COMPUTER SCIENCE

COS 222 - Practical 4

15043143, Regan Koopmans

October 26, 2016

1 INTRODUCTION

For this practical we had to compare memory mapped I/O with standard file I/O in terms of read speed. For testing this I wrote two C programs, each using one of the two methods, and tested each method using variety of files and parameters.

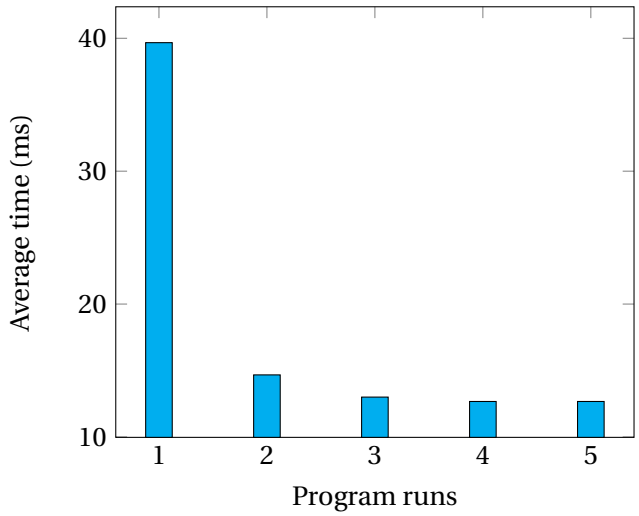
2 FINDINGS

Each of the five cases were run **3 times**, on each 3 different files. An average was then generated from these runs and were used as ordinal values amongst the program runs. These values can be seen in the **table** and **histograms** below:

2.1 FILE 1

Table 2.1: File 1 Results

Column Number	*Program Run*	*Average Time (ms)*
1	Task 1 - Chunk 1	39.67
2	Task 1 - Chunk 32	14.67
3	Task 1 - Chunk Default (1024)	13
4	Task 1 - Chunk Max (8192)	12.67
5	Task 2	12.67



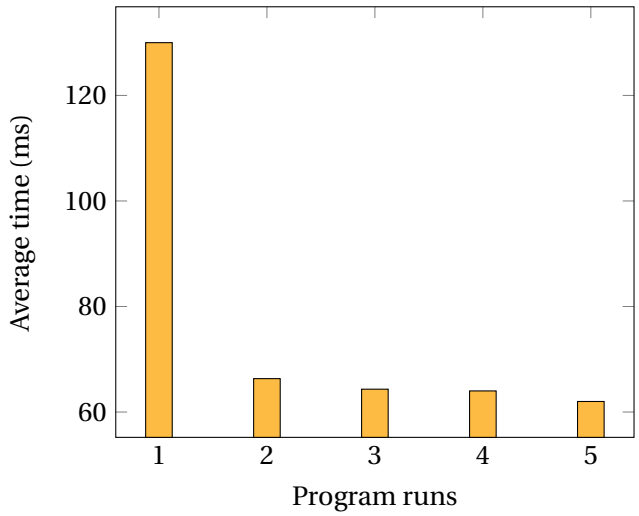
The fastest of these runs was: Task 1 with chunk size of 8192, **and** Task 2 (tie).
The slowest of these runs was: Task 1 with chunk size of 1.

STATISTICAL SIGNIFICANCE: For this particular file, there was not a tremendous **numerical difference** between the slowest and the fastest, a value of only 27 ms. However, the slowest program ran **325% slower** than the fastest, which is quite signficant.

2.2 FILE 2

Table 2.2: File 2 Results

Column Number	*Program Run*	*Average Time (ms)*
1	Task 1 - Chunk 1	130
2	Task 1 - Chunk 32	66.33
3	Task 1 - Chunk Default (1024)	64.33
4	Task 1 - Chunk Max (8192)	64
5	Task 2	62



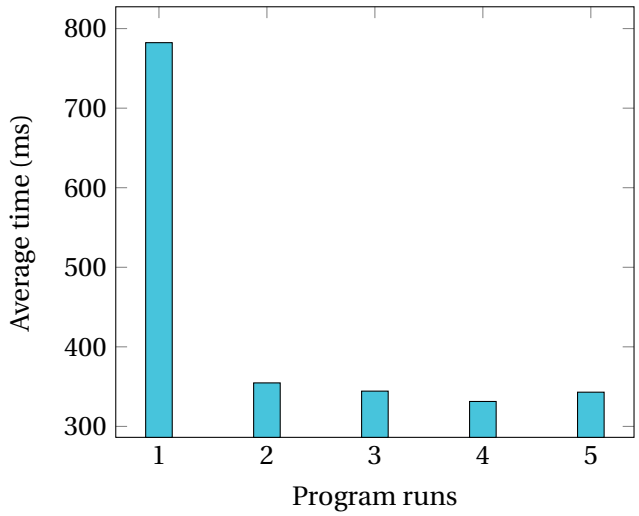
The fastest of these runs was: Task 1 with chunk size of 8192.
The slowest of these runs was: Task 1 with chunk size of 1.

STATISTICAL SIGNIFICANCE: For this particular file the difference between the slowest and fastest run was 68 ms. The slowest program ran about **210% slower** than the fastest run. This is just over double the time, and while less than in File 1, it is still quite substantial.

2.3 FILE 3

Table 2.3: File 3 Results

Column Number	*Program Run*	*Average Time (ms)*
1	Task 1 - Chunk 1	782.33
2	Task 1 - Chunk 32	354.67
3	Task 1 - Chunk Default (1024)	344.33
4	Task 1 - Chunk Max (8192)	331.33
5	Task 2	343



The fastest of these runs was: Task 1 with chunk size of 8192.
The slowest of these runs was: Task 1 with chunk size of 1.

STATISTICAL SIGNIFICANCE: For this particular file the difference between the slowest and fastest runs was 451 ms, which is an impressively large difference. Task 1 with a chunk size of 1 took more than $\frac{3}{4}$ seconds. The slowest run, ran approximately **228% slower** than the fastest run.

3 CONCLUSION

3.1 OBSERVATION

Surprisingly, the mmaped program seems, in general, to run very slightly slower than the standard I/O with the largest chunk size, which was not my initial expectation. This may be because as chunk sizes become close to or greater than the actual file size, **standard I/O functions approximate the behaviour of memory mapped I/O**. A more concrete investigation would have to be made to confirm this.