Andrew Nolan

CS 3013

Project 4

10/11/18

Project 4 Statistical Analysis

Project 4 involved various methods of searching for strings in files. We used the read system call on chunks, memory mapping, and multi-threaded memory mapping. The following graphs compares the wall clock times of the various methods. Unfortunately, on my virtual machine no major page faults were recorded. My machine is using 2 cpus.

The methods tested are, read with chunk sizes: 1, 1K, 4K, and 8K, a single threaded memory map, as well as memory maps with 2, 4, 8, and 16 parallel threads.

The search string remained constant between all files. The word “test” was searched for.

The files tested varied in size and type. A 13.7 KB executable file, a 3.1 KB text file, a 7.4 KB .cpp file, a 182.7 KB .bin file, a 6.8 MB executable, a 1.2 MB change log, and a 1.1 MB log file.

Here is a table of the data produced for each method, file sizes are in KB and time is in milliseconds:



The graph produced was messy due to the high times of the 1 byte per chunk search, so I included that graph and a graph not including the 1-byte data. The 1-byte data goes so high that all the other trendlines overlap each other and appear flat. The graphs can be seen on the next page.

Based on this information we can see that all the methods have a relatively similar performance at low file sizes. Around 1 MB, the 1-byte search becomes very slow. At even larger file sizes the memory mapped approaches, even when multi-threaded appear to perform slower than the reading chunk approaches. With the 4 KB and 8 KB approaches being the best overall, never going longer than half of a second. At least based on the data below. On a different machine these results may be different.