COMP20010 Lab Six: Algorithm Analysis

Thomas Swann

December 8, 2015

1 Part 1 Algorithm

1.1 Asymptotic run-time analysis

My algorithm runs in asymptotic time $O(wN)$. The argument for this follows:

In my program I implemented the Radix Sort algorithm. Radix Sort is a non-comparitive integer sorting algorithm that sorts the data by grouping the elements with the same significant position. After grouping the elements it uses a bucket to reorder the elemements into the correct group. It does this as many times as the longest element. This means that the time complexity is $O(wN)$ where w is the length of the largest integer. This is because it does w passes and in each pass it looks at all N elements.

1.2 Experiments

For my experiments I used the time command for each size of datafile. I then put this data in a data.dat file and used the gnuplot scripts given to plot a graph and fit an equation to the data. I also used "fitscript.p" to find a possible equation to test against. Here are the results:

data size	run time	$t = 4.39 \times 10^{-7} \times N$
10	0.001s	0.00000439s
100	0.001s	0.0000439s
1000	0.002s	0.000439s
10000	0.006s	0.00439s
100000	0.046s	0.0439s
1000000	0.441s	0.439s
2000000	0.878s	0.878s
3000000	1.344s	1.317s
4000000	1.748s	1.756s
5000000	2.206s	2.195s
6000000	2.622s	2.634s
7000000	3.106s	3.073s
8000000	3.503s	3.512s
9000000	3.953s	3.951s
10000000	4.379s	4.390s

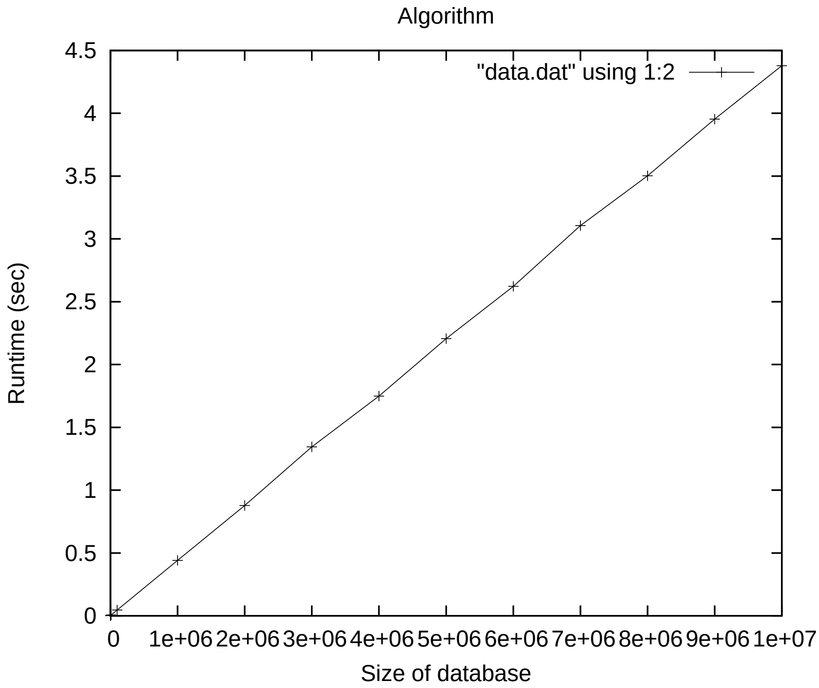

Figure 1: Plot of my runtimes

1.3 Prediction

My estimate of the equation for the run-time of the algorithm is:

$$t(N) = 4.39 \times 10^{-7} \times N \tag{1}$$

Using this, the estimated time to find the ninetieth percentile of a file containing 60 million numbers is 26.34 seconds. After running the program with a datafile of size 60 million the time recorded was 26.262s which confirms my prediction.

2 Part 2 Algorithm

2.1 Asymptotic run-time analysis

My algorithm runs in asymptotic time $O(wN)$.

The argument for this follows: As my program before had a time complexity of $O(wN)$ I didn't change the algorithm used, therefore my reasoning is the same. I did improve the program to make it more efficient. Using the knowledge of how big the numbers will be I removed the part of the program that finds the largest integer and instead did a known number of passes. This obviously will slow the program down for small numbers of N however when doing the experiments the program executed too fast to notice.

2.2 Experiments

I tested the program using the same method as before, however when I plotted the graph I used the "plot2data.p" script (data1.dat is the more efficient program and data2.dat is the previous program). Here are the results:

data size	run time	$t = 4.29 \times 10^{-7} \times N$
10	0.001s	0.00000429s
100	0.001s	0.0000429s
1000	0.002s	0.000429s
10000	0.006s	0.00429s
100000	0.045s	0.0429s
1000000	0.433s	0.429s
2000000	0.863s	0.858s
3000000	1.289s	1.287s
4000000	1.729s	1.716s
5000000	2.148s	2.145s
6000000	2.584s	2.574s
7000000	3.011s	3.003s
8000000	3.429s	3.432s
9000000	3.865s	3.861s
10000000	4.282s	4.290s

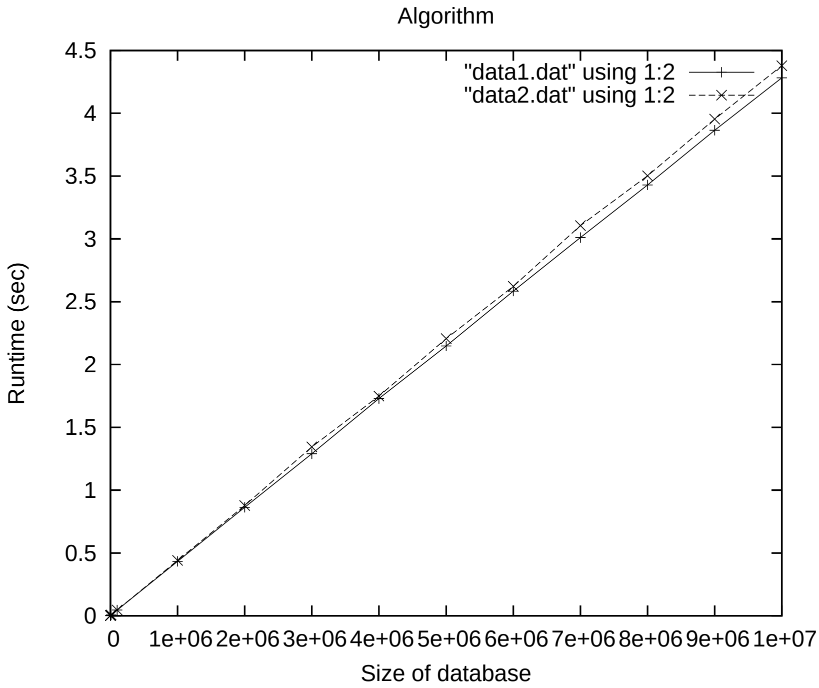

Figure 2: A comparison of both programs

2.3 Prediction

My estimate of the equation for the run-time of the algorithm is:

$$t(N) = 4.29 \times 10^{-7} \times N \tag{2}$$

Using this, the estimated time to find the ninetieth percentile of a file containing 60 million numbers is 25.74 seconds. After running the program with a datafile of size 60 million the time recorded was 25.770s which confirms my prediction.