Estimate Secret Algorithms Assignment

Tom Bohbot

September 2020

1 Introduction

I began this homework assignment through coding an inner class called Stopwatch which times how long a program takes to run. Then I made methods similar to Sedgewick's doubling method for each secret algorithm and implemented the Stopwatch to time each secret method depending on the input inserted.
Once I began to run the methods I ran the methods until they began to take very long to return. I repeated this for each algorithm five times and then took the average of those numbers and inserted them into the tables below.
Afterwards, I produces a log-log chart with all of the algorithms together to be able to compare their runtimes.
Using all this information I was able to estimate their runtime without looking at the source code.

2 SecretAlgorithm1

n times ran	time taken(ms)	ratio
128	0.003	3.000
256	0.006	2.000
512	0.015	2.500
1,024	0.100	6.667
2,048	0.904	9.040
4,096	17.619	19.490
8,192	137.193	7.787

NOTE: Data entry before 128 averaged zero milliseconds to run.

3 SecretAlgorithm2

n times ran	time taken(ms)	ratio
2,097,152	0.001	1.000
4,194,304	0.002	2.000
8,388,608	0.002	1.000
16,777,216	0.005	2.500
33,554,432	0.008	1.600
67,108,864	0.047	5.880
134,217,728	0.075	1.600
2,684,354,568	0.156	2.080

NOTE: Data entry before 2,097,152 averaged zero milliseconds to run.

4 SecretAlgorithm3

n times ran	time taken(ms)	ratio
512	0.001	1.00
1,024	0.003	3.00
2,048	0.002	0.67
4,096	0.008	4.00
8,192	0.013	1.63
16,384	0.051	3.92
32,768	0.170	3.33
65,536	0.642	3.78
131,072	10.724	16.70
262,1444	8.714	0.81
524,288	35.628	4.09

NOTE: Data entry before 2,097,152 averaged zero milliseconds to run.

5 SecretAlgorithm4

n times ran	time taken(ms)	ratio
32,768	0.003	3.00
65,536	0.007	2.33
131,072	0.036	5.14
262,1444	0.045	1.25
524,288	0.079	1.75
67108864	0.216	2.73
2,097,152	0.421	1.95
4,194,304	0.759	1.80
8,388,608	1.733	2.28
16,777,216	3.450	1.99
33,554,432	7.233	2.10
67,108,864	15.202	2.10
134,217,728	31.506	2.01

NOTE: Data entry before 32,768 averaged zero milliseconds to run.

6 Log-Log Plot

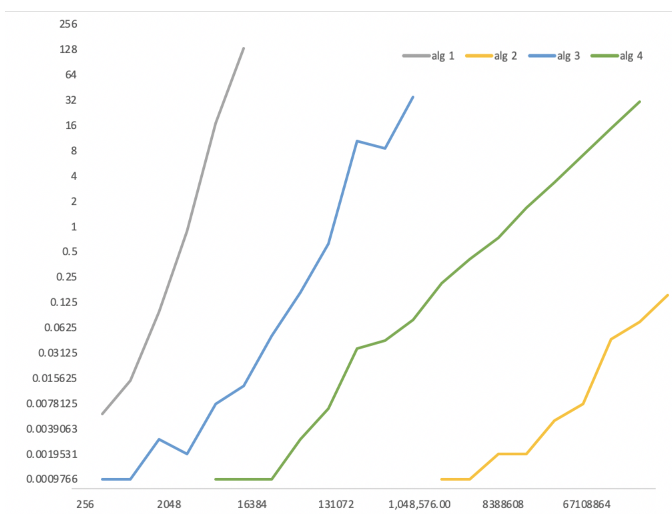

Figure 1: Log-Log Plot of Secret Algorithms 1 - 4

NOTE: The absence of line at the beginning of each line means that the runtime was zero for the corresponding input.

7 Estimating Runtimes

Secret Algorithm 1:
I believe that Secret Algorithm 1's run time is O(n^3). The ratio for this algorithm seems to approach 8 if one disregards the outlier of 21.907. Coupling the ratio approaching eight with the illustration provided by the log-log plot shows that this algorithm has a runtime of O(n^3).
Secret Algorithm 2:
I believe that Secret Algorithm 1's run time is O(n). The ratio of this algorithm approaches 2 which is normal for an O(n) runtime. Additionally, the log-log plot illustrates that it's slope is most similar to an O(n) runtime.
Secret Algorithm 3:
I believe that Secret Algorithm 1's run time is O(n^2). The ratio of this algorithm approaches 4 if we disregard the outliers, which is a normal ratio for this runtime. Additionally, its slope grows at an O(n^2) rate as provided through the illustation of the graph.
Secret Algorithm 4:
I believe that Secret Algorithm 1's run time is O(nlogn). I belive that this runtime is very similar to O(n) when only looking at the ratios as it approaches a limit slightly above two, but not so high to consider O(n^2). However, when looking at the graph, it is obviously slower than algorithm 2 which is O(n) and considerably quicker than the O(n^2) runtime, which places this alogrithm perfectly to be considered O(nlogn).