EECS 233 HW1

Benjamin Pierce (bgp12)

9/4/17

1 Questions

1.1 Question 1

1.1.1 Run Time Less Then 1s

Fig. 1.1.1: Algorithim 1

N = 0	Time Elapsed: 0ms
N = 10000000	Time Elapsed: 6ms
N = 20000000	Time Elapsed: 7ms
N = 30000000	Time Elapsed: 11ms
N = 40000000	Time Elapsed: 14ms
N = 50000000	Time Elapsed: 15ms
N = 60000000	Time Elapsed: 20ms

Fig. 1.1.2: Algorithm 2

N = 0	Time Elapsed: 0ms
N = 1000	Time Elapsed: 3ms
N = 2000	Time Elapsed: 3ms
N = 3000	Time Elapsed: 3ms
N = 4000	Time Elapsed: 7ms
N = 5000	Time Elapsed: 9ms
N = 6000	Time Elapsed: 12ms

Fig. 1.1.3: Algorithm 3

N = 0	Time Elapsed: 0ms
N = 1000	Time Elapsed: 3ms
N = 2000	Time Elapsed: 2ms
N = 3000	Time Elapsed: 3ms
N = 4000	Time Elapsed: 5ms
N = 5000	Time Elapsed: 7ms
N = 6000	Time Elapsed: 12ms

1.1.2 Run Time Greater then 1s

Fig. 1.2.1: Algorithim 1

N = 0	Time Elapsed: 0ms
N = 10000000000	Time Elapsed: 3194ms
N = 20000000000	Time Elapsed: 6440ms
N = 30000000000	Time Elapsed: 9418ms
N = 40000000000	Time Elapsed: 12536ms
N = 50000000000	Time Elapsed: 15291ms
N = 60000000000	Time Elapsed: 18557ms

Fig. 1.2.2: Algorithim 2

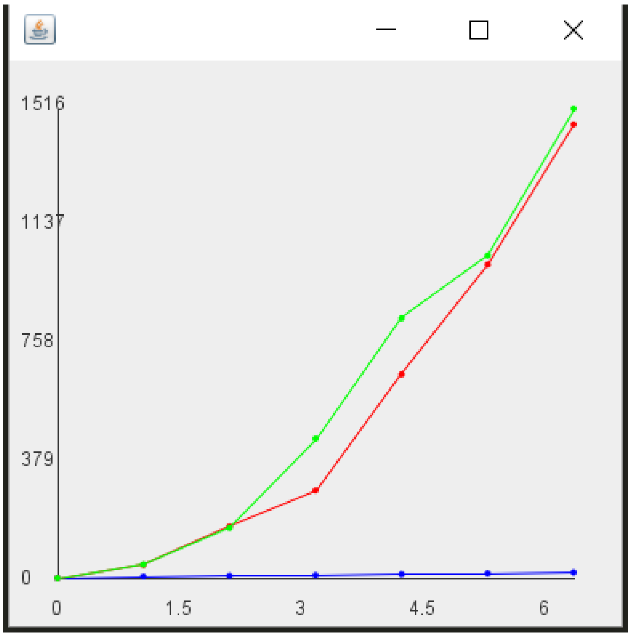
N = 0	Time Elapsed: 0ms
N = 100000	Time Elapsed: 4098ms
N = 200000	Time Elapsed: 17206ms
N = 300000	Time Elapsed: 27928ms
N = 400000	Time Elapsed: 51137ms
N = 500000	Time Elapsed: 80322ms
N = 600000	Time Elapsed: 118394ms

Fig 1.2.3: Algorithm 3

N = 0	Time Elapsed: 0ms
N = 100000	Time Elapsed: 4098ms
N = 200000	Time Elapsed: 18293ms
N = 300000	Time Elapsed: 29799ms
N = 400000	Time Elapsed: 54061ms
N = 500000	Time Elapsed: 80339ms
N = 600000	Time Elapsed: 111339ms

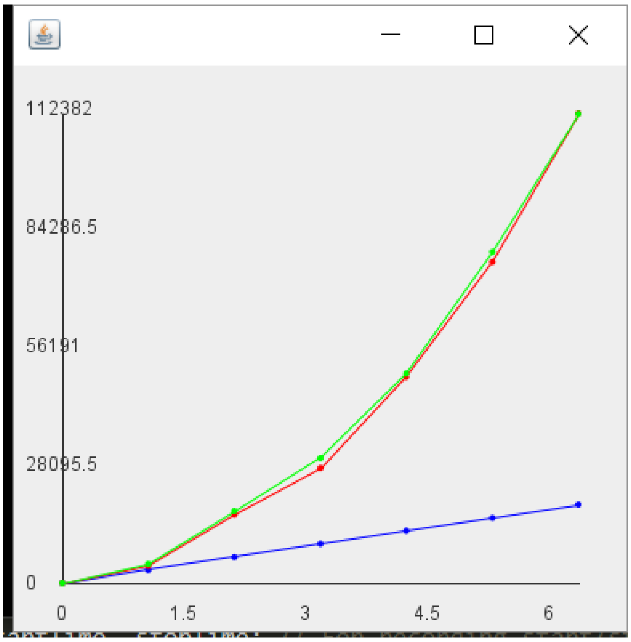
1.2 Question 2

1.2.1 Run Time Less Then 1s



Blue:Algorithm 1; Red:Algorithm 2; Green:Algorithm 3

1.2.2 Run Time Greater Then 1s



Blue:Algorithm 1; Red:Algorithm 2; Green:Algorithm 3

1.3 Question 3

The results appear to be roughly as expected. The first algorithm executes a single *for* loop, as as such run in $O(n)$ time. The second algorithm executes nested *for* loops; therefore, it runs in $O(n^2)$ time. The third algorithm executes a single *for* loop, and then a nested loop, giving it a designation of $O(n^2)$. In the results, the first algorithm increases linearly for increasing n, as expected with an $O(n)$ algorithm. The second algorithm appears to increase quadratically as expected. The third algorithim also increases quadratically, with slight interference at small n from the initial *for* loop.

1.4 Question 4

In the first algorithm, there is little diffrence for large or small n, due to the algorithm's linearity. Likewise, the second algorithm behaves as it did with small n. However, the third algorithm shows a different result; as n increases, the graph becomes more quadratic. This is because of the dominance of the polynomial term n^2 as the n time delay caused by the first *for* loop becomes less important as the n^2 term become far larger. Run times longer then one second can have multiple uses, such as in in paralell processing when it becomes important that two cores operate in tandem; in the case that a core falls behind, the other must take extra time so that it does not attempt to process data that doesn't exist.