CS345A Special Assignment Report

Ashish Kumar Singh(14142) , Shubham Prakash(14683)

October 15, 2017

Plot of CPU time vs n for both the algorithms

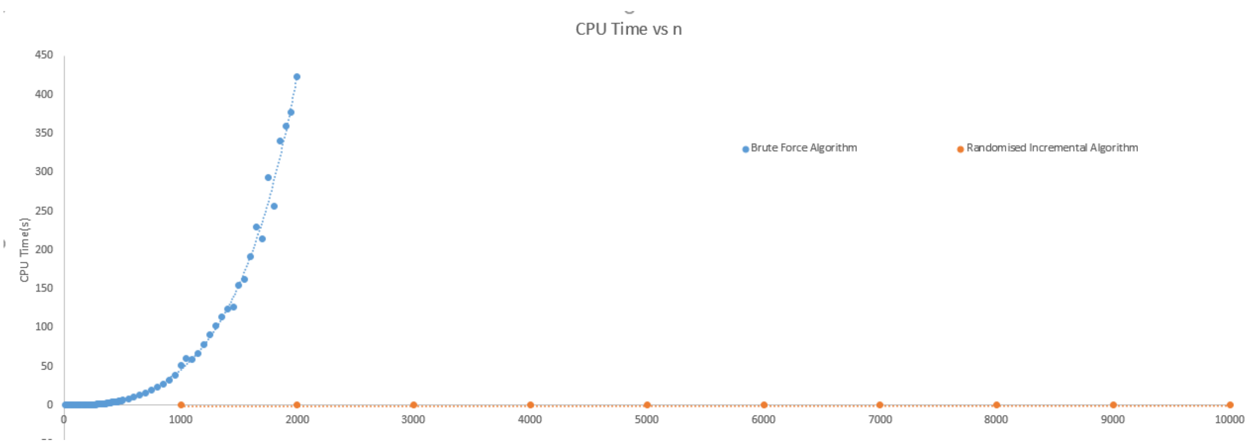

Figure 1: CPU Time vs n. (blue) Brute force method (red) Randomized Incremental Algorithm

log-log graph property: Slope in log-log graph will give actual exponent.
Proof: Let $y = cx^k$, if we plot $log(y)$ vs $log(x)$ the claim is that the slope will be k.

$$slope = \frac{d(log(y))}{d(log(x))} = \frac{x}{y}\frac{dy}{dx} = \frac{x}{y}ckx^{k-1} = \frac{ckx^k}{cx^k} = k$$

Plot of $log(CPUtime)$ vs $log(n)$ for both Algorithms

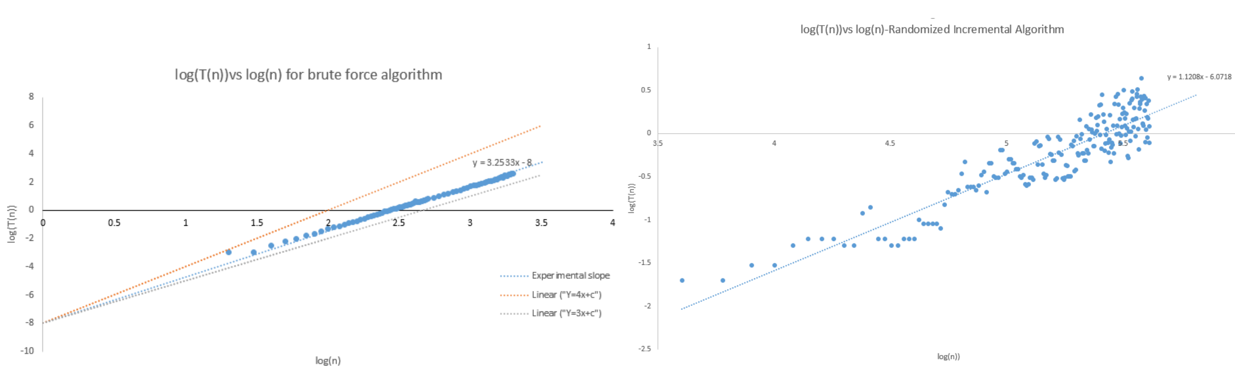

Figure 2: $log(CPUtime)$ vs $log(n)$. (left) Brute force method (right) Randomized Incr. Algorithm

Inferences

- From Figure1 we can easily see that running time of Brute Force algorithm is of much higher order than that of Randomized Incremental Algorithm.
- In Figure2(left), the best linear fit was with $y = 3.2533x - 8$, which is lower bounded by $y = 4x - 8$and upper bounded by $y = 3x - 8$ thus empirically, the average(expected) running time of brute force algorithm came out to be $n^{3.2533}$ which is lower bounded by $O(n^4)$.
  Note: Due to time constraints, the algorithm was ran till $n = 2000$ points thus empirical average of running time of $n^{3.2533}$ is only for $n \leq 2000$.
- In Figure2(right), the best linear fit was with $y = 1.1208x - 6.0718$, thus from our data the empirical expected running time of Randomized Incremental Algorithm is $O(n^{1.1208})$.

Distribution of running(CPU) time of Randomized Incremental Algorithm

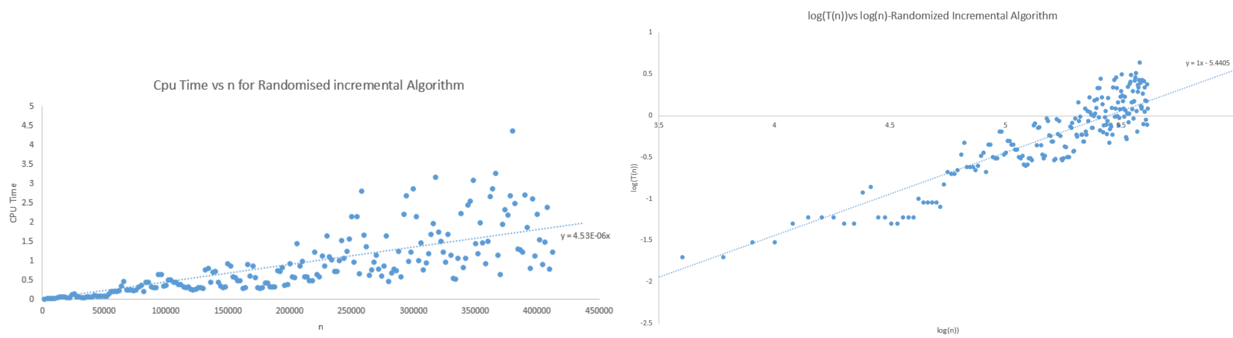

Figure 3: (left) CPU time vs n with linear regression fit. (right) $log(CPUtime)$ vs $log(n)$ with best fit with slope 1 for Randomized Incremental Algorithm

- Figure3(left) shows the distribution of running(CPU) time vs n for randomized algorithm. We can see that for large n there is large deviation in running time, but running time seems to be increasing almost linearly $O(n^{1.1208})$.
- Figure3(right) shows the log-log plot with best linear fit with slope=1. Thus, any point lying below this line will be O(n). On the other hand, points lying above will not have O(n) as their lower bound, hence those are the points that deviate from O(n). In the experiment 83 points out of 206 data point deviated from O(n) bound i.e. approximately 40% data points deviated from O(n) lower bound.