Cs301,Spring 2020

Assignment 1

Yunus Emre TAT 25187

Problem 1

a)T(n)=2T(n/2)+n^3 , from master theorem case 3 will be occured,

$f(n) = \Omega(n^{log_b^a+e})$ for some $e > 0$ and if af(n/b)<=cf(n) for some c<1 then T(n)= $\Theta(f(n))$

So that, $n^3 = \Omega(n^{2+\log_2 2})$ so,T(n)= $\Theta(n^3)$

b)T(n)=7T(n/2) +n^2 , from master theorem case 1 will be occurred,

$f(n) = O(n^{log_b^a-e})$ for some e>0 then T(n)= $\Theta(n^{\log_b a})$

So that, $n^2 = \Omega(n^{\log_2 7-e}$) e is bigger than 0 so it is T(n) = $\Theta(n^{log_2^7})$

c)T(n)=2T(n/4)+$\sqrt{n}$,from master theorem case 2 will be occurred,

$f(n) = \Theta(n^{log_b^a})$ then T(n)= $\Theta(n^{log_b^a} \log n)$

So that, $\sqrt{n} = \Theta(n^{log_4^2})$ so,$T(n) = \Theta(\sqrt{n}\ \log n)$

d)T(n)=T(n-1)+n,

T(n-1)=T(n-2)+n-1

.

.

.

T(n-k+1)=T(n-k)+n-k+1 sum the left hand side and right hand side end of the matematical substraction,

For k=n,T(n)=T(n-k)+$\sum_{k=0}^{n-1} n - k$

T(n)=T(0)+(n(n+1))/2,in this situation T(0) is constant so that T(n)= O(n^2)

Problem 2

A)

i) In Naive recursion algorithm ,We are trying to generate all subsequences of both given sequences so that we are getting same datas which are called as overlapping.It is the calculating same things everytime.This solution is exponential in term of time complexity.

There are two string which their lengths are m and n

If m equals n,The recurion tree nodes have two branch and function will be like this,T(n)=2T(n-1),T(n)=2^2T(n-2) it will go like this and so that T(n)=2^kT(n-k) if n=k so that,T(n)=2^n if they are not equals if m>n it will be T(n)=2^m means O(2^n)

ii)Overlapping substructure property is avoided by the Memoization.We are trying to store the recursion in matrix.It will prevent calculating the same datas which we calculated.It provides utilizing the algorithm.We will have already in the table so that memorization reduces number of function call.Total how many calls were made depens number of elements in the table.Complexty is O(mxn) which memorization matrix's area are m is the length of first string parameter, n is second string parameter.

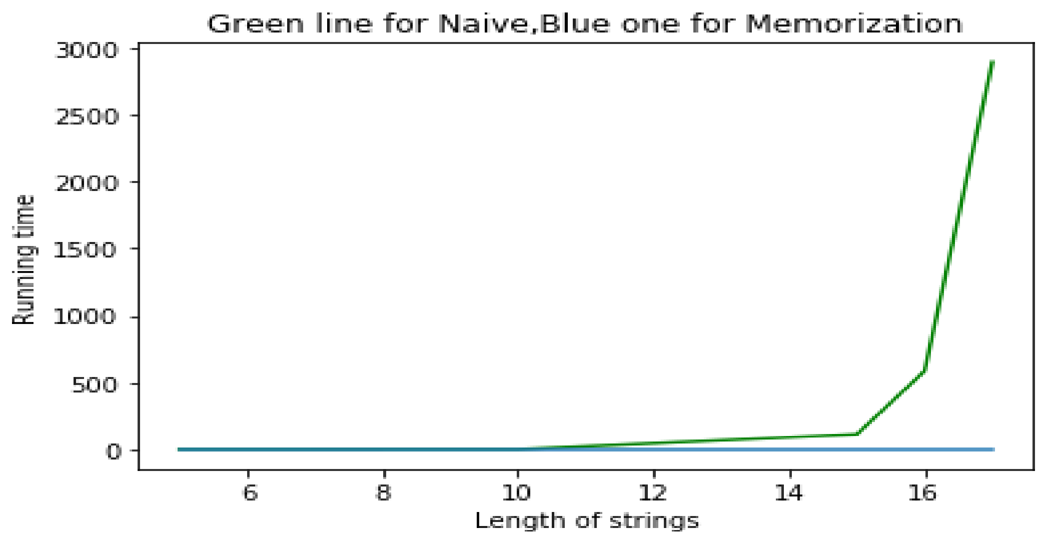
B)

I)

Algorithm	m=n=5	m=n=10	m=n=15	m=n=16	m=n=17
Naïve	0.0003001089	0.27445564270	112.452389742	584.84723985	2894.5823094
Memorization	0.00003140724512	0.0002232495561400	0.0004996040	0.0006012855	0.00072286

Operating system=Windows 8 ,Ram=8gb

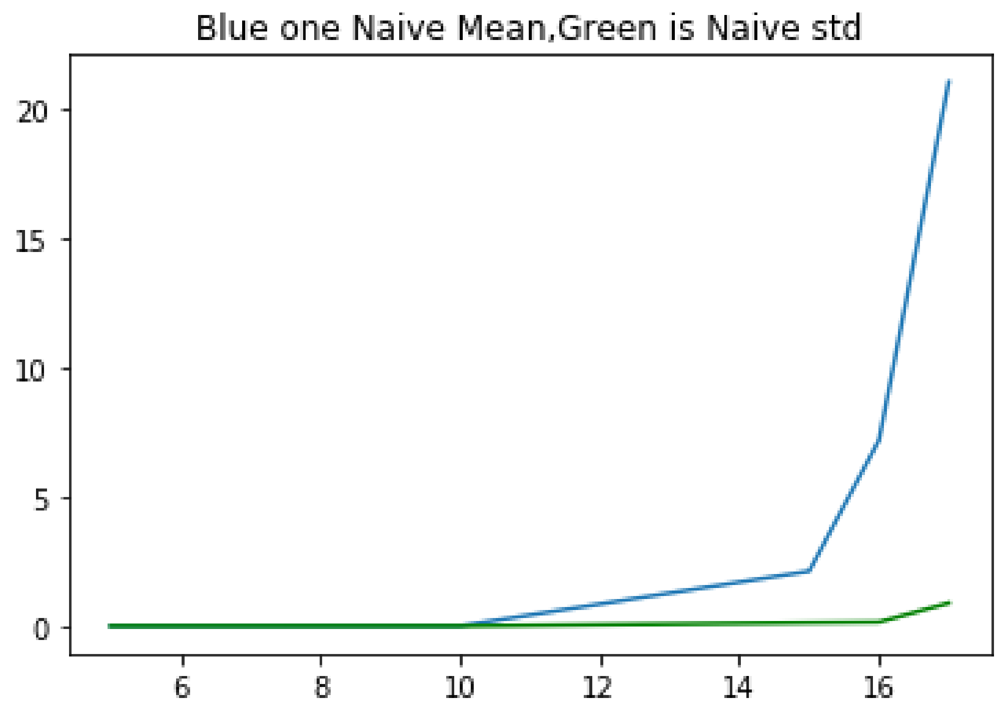
ii)

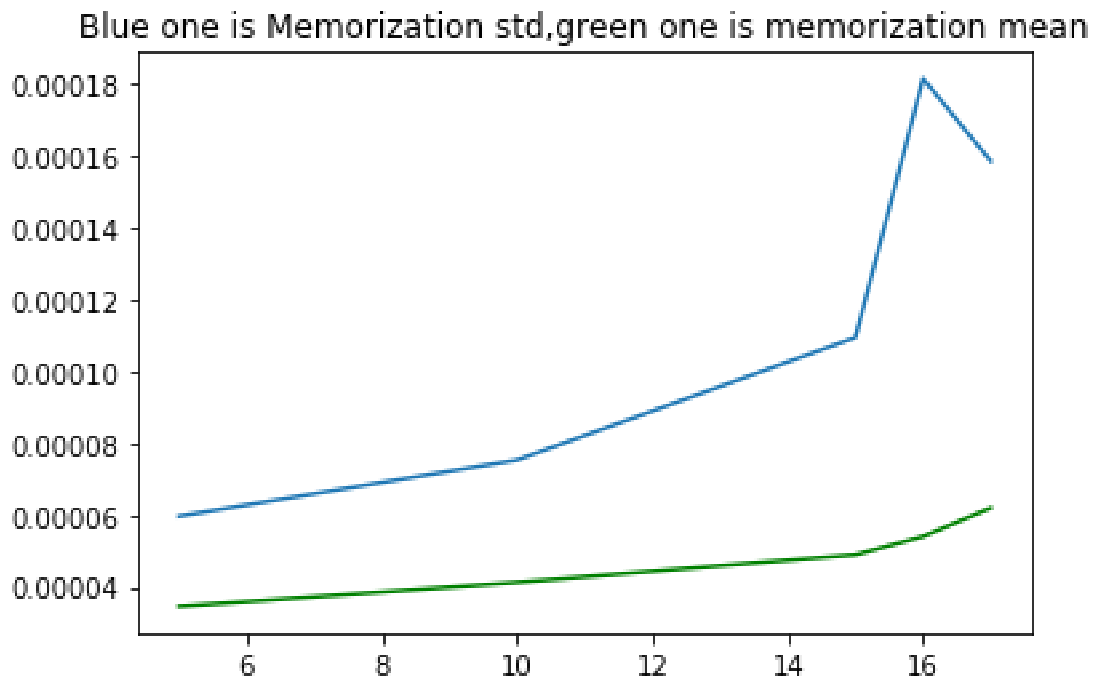


iii)Yes experimantal results confirm the theoretical result which were one of them is Naive algoritms best asymp. worst case running is exponentially.As we see in the b-ii part green line which is exponentially growing.On the other hand,Memorization algorithms is O(mxn) was increasing linearly.Experimental result is as same as theoritical result. Naive algorithm will spend more time to compile with a big difference.

C)

Algorithm	m=n=5		m=n=10		m=n=15		m=n=16		m=n=17	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Naive	0.0000913967269	0.000017894352	0.004502135	0.0013964382	2.12493558208	0.1359825	7.185982985	0.09576823	21.023858925	0.8928583
Memoization	0.00003058742	0.0000495863	0.0000412453	0.000045523845	0.00004885679250	0.00005937593	0.00005395720	0.000071285214	0.000061917593	0.00006846385





iiii) The string parameters which we generated from the random DNA sequence are random variables so that they are not the worst case because they are not different completely between each other.But in general when we look their graphs their shapes are almost same because algorithms complexity are not changed it is still same .Standard deviations shows us there is no big difference between DNA sequence especially in the Naive algorithm.