SUMMARY

USC ID/s: 1487027774 and 9495284506

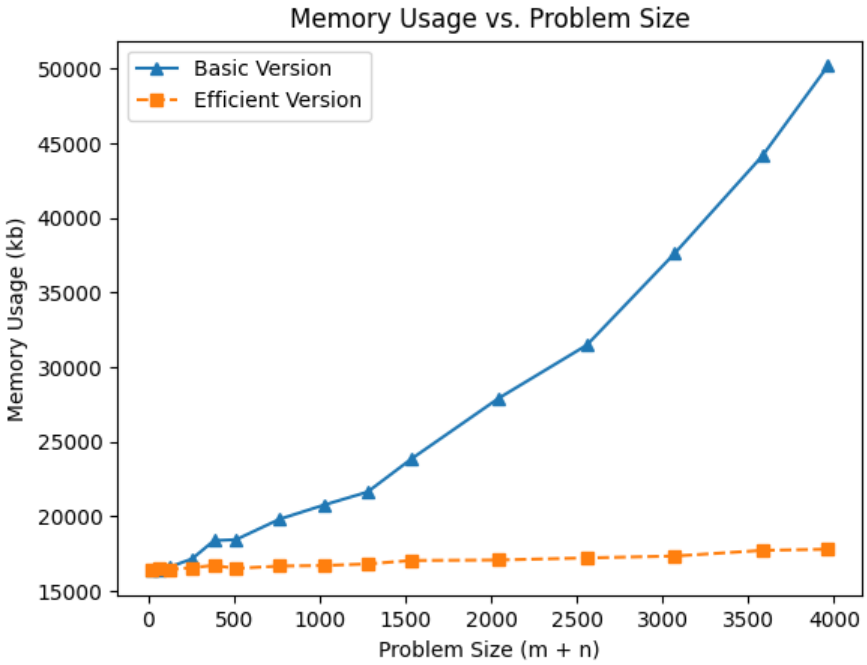
Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.0	1.07	16376	16328
64	1.018	1.023	16388	16464
128	4.28	5.404	16616	16468
256	1.017	4.538	17064	16536
384	17.07	23.36	18032	16692
512	20.127	68.41	18676	16584
768	71.601	126.05	18720	16640
1024	120.79	247.812	18716	16712
1280	207.245	396.50	18420	16708
1536	310.18	580.53	19528	17044
2048	547.69	1051.90	19708	17004
2560	847.061	1595.63	18728	17316
3072	937.46	1839.109	18948	17388
3584	1369.57	2334.42	19388	17540
3968	1688.89	3243.93	19920	17660

Insights

Basic Dp approach on solving string matching is computationally efficient when compared to space efficient version. But since in case of DNA sequencing the memory required can go upto billions, we accept the trade-off between computationally efficient and spatially efficient approaches. In following graphs, we will discuss this in detail.

Graph1 – Memory vs Problem Size (M+N)



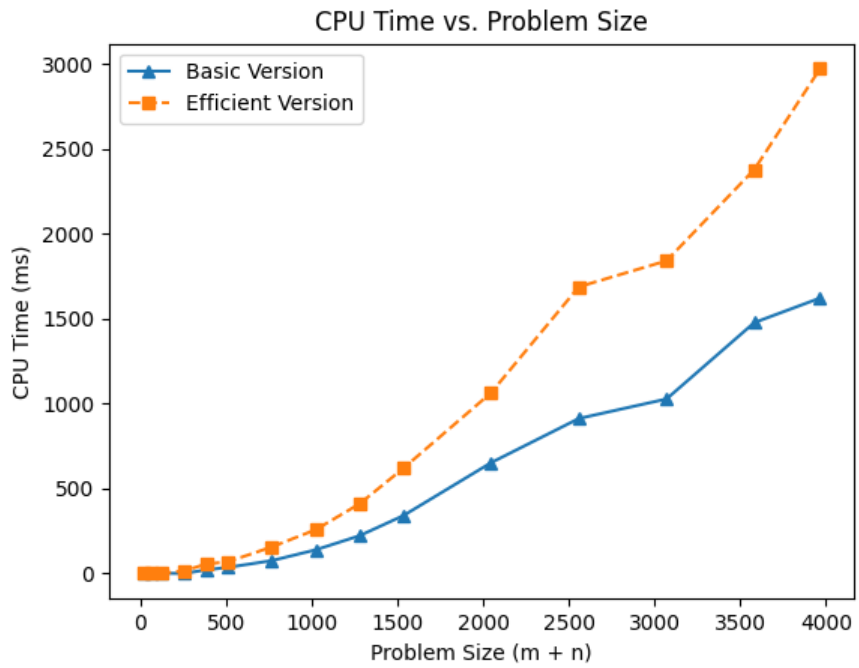
Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)

Basic: Memory increases polynomially as the size of the input increases.
Efficient: Memory usage is linear in nature.

Explanation:

In Basic DP approach while computing the value of the optimal solution, we use space of size O(m*n) where m and n are the sizes of input string X and Y. Thus as the input size (m+n) increases, memory consumed increases polynomially.
In Efficient approach we use divide and conquer for string matching, we divide one of the input strings midway and find the division point for other which gives the total lowest cost. Due to this approach, as we keep dividing the input repeatedly, and as we need to access only two columns at a time, we end up using O(n+m) auxiliary space which is linear.
So, we can say that basic approach has worse space complexity than efficient.

Graph2 – Time vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)

Basic: Time consumed increases polynomially as the input size increases.
Efficient: There is a polynomial increase in time consumed as the input size increases.

Explanation:

The graph shows how both approach's time consumed is affected as size of the input increases. In the Basic DP approach, we use top-down pass to compute the value of optimal solution and in the bottom up pass we compute the actual string matching. The time complexity of this approach is O(m*n). In efficient divide and conquer we repeatedly divide the input string and in a single divide and conquer iteration find the string matching. This approach takes polynomial time i.e. O(m*n) as well. Although both are polynomial, in efficient approach the number of calculations is significant thus it tends to take more time.

Contribution

1487027774 and 9495284506 : Equal Contribution