SUMMARY

USC ID/s: 9391074897, 1556486977, 71353211165

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.552	0.524	2.41254e+06	2.4576e+06
64	1.679	1.427	2.43712e+06	2.44531e+06
128	1.994	4.309	2.44531e+06	2.48627e+06
256	7.804	15.524	2.53952e+06	2.51085e+06
384	16.935	34.071	2.62554e+06	2.51904e+06
512	32.666	63.616	2.90816e+06	2.51904e+06
768	78.872	159.68	3.30138e+06	2.51904e+06
1024	121.925	289.436	3.87891e+06	2.71155e+06
1280	202.188	425.173	4.57523e+06	2.60506e+06
1536	257.155	547.658	5.26746e+06	2.51085e+06
2048	481.128	1014.03	7.25402e+06	2.7648e+06
2560	481.128	1542.75	9.76486e+06	2.81805e+06
3072	997.287	2082.16	1.28369e+07	2.80576e+06
3584	1367.4	3077.82	1.63963e+07	2.6624e+06
3968	1659.12	3551.47	1.89358e+07	2.82214e+06

Datapoints

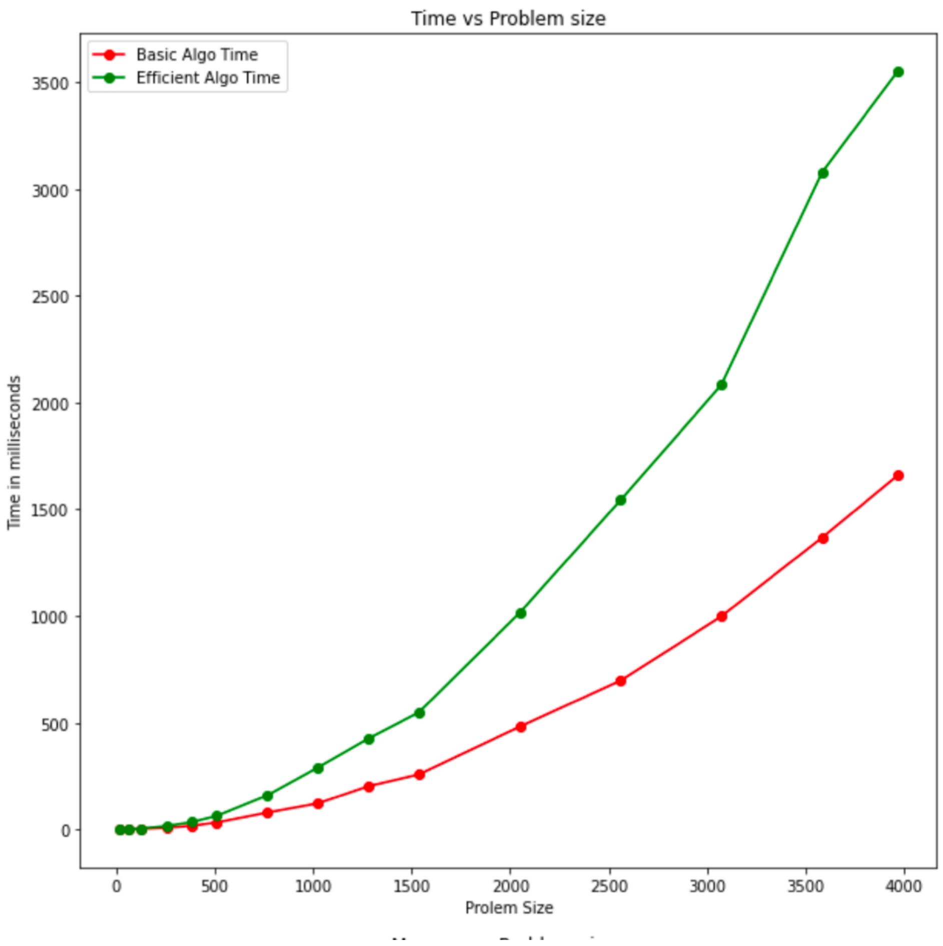
Insights

For Sequence Alignment, we implemented two algorithms. First approach towards the problem was using dynamic Programming. The algorithm demands a heavy usage of memory. However, we know that for the calculation of optimal cost, the only requirement is the immediate previous output rather than all the previous outputs. Hence efficient version of algorithm generated using divide and conquer, will only store the immediate previous output at every step which will eventually reduced our memory usage.

The substantial difference can be seen in memory requirement as the input size increases.

Following graphs depicts the difference between the time and memory requirements in both algorithm.

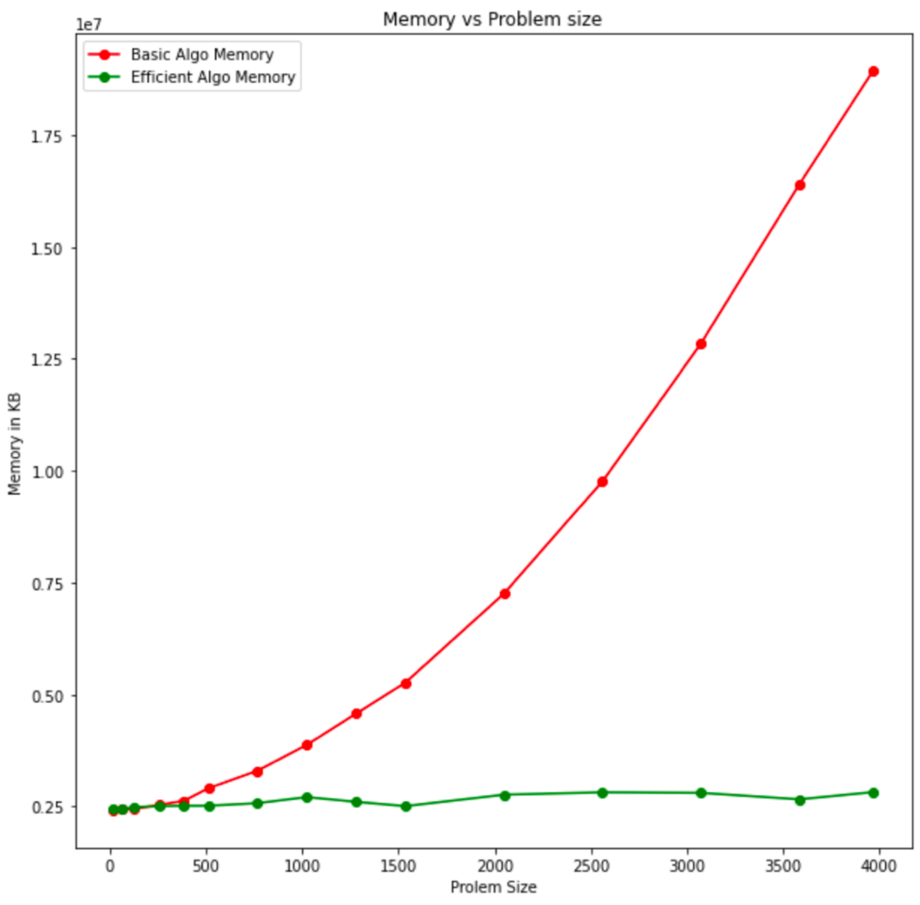
Graph1 – Time vs Problem Size (M+N)



Basic: Graph for basic algorithm takes O(m*n) time. So the nature of the graph is polynomial.
Efficient: Graph for efficient algorithm takes O(m * n) time. So the nature of the graph is polynomial.

Explanation:
Both algorithms work on m*n size workspace, so they take polynomial time. However, efficient algorithm takes more time for larger sized inputs because it runs recursively.

Graph2 – Memory vs Problem Size (M+N)



Basic: Graph for basic algorithm takes O(m*n) memory. So the nature of the graph is polynomial.
Efficient: Graph for efficient algorithm takes O(m + n) memory. So the nature of the graph is linear.

Explanation:

As we use recursive functions in efficient algorithm, we are reusing the workspace. Hence reducing memory usage. This difference in memory of both algorithms is visible as the input size increases.

Contribution

9391074897 : Equal Contribution
1556486977 : Equal Contribution
7135321116 : Equal Contribution