SUMMARY

USC ID/s:

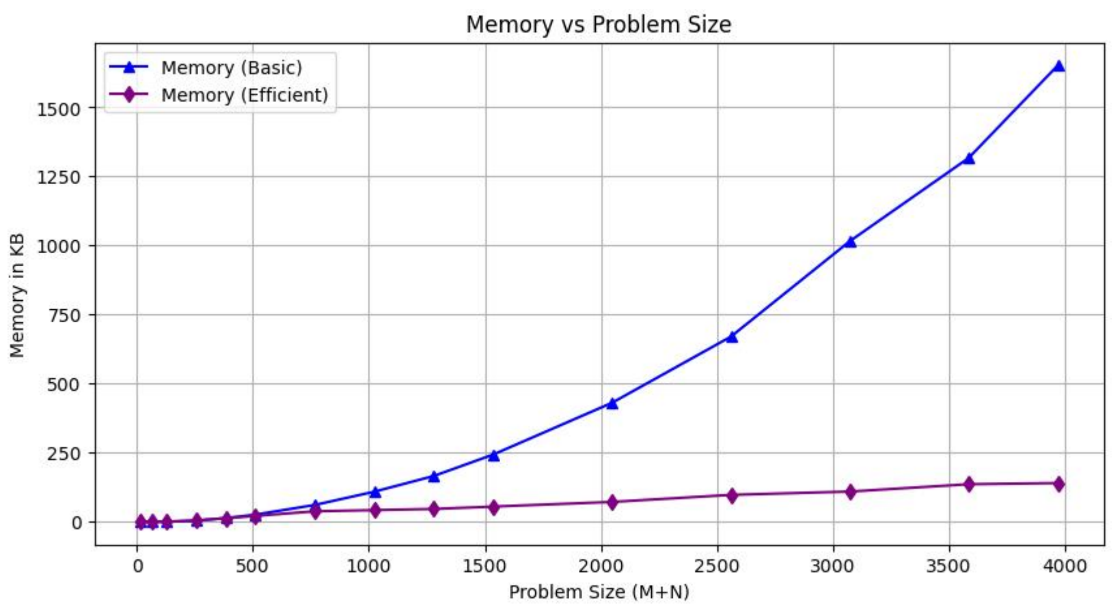
9015551525_4675328960_8157970159

Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.0042000003	0.09429	0.0	0.0
64	0.029029999	0.108219996	0.0	0.0
128	0.09124	0.27375	0.0	0.0
256	0.31559	0.73059	4.2472	4.2272
384	0.62227	1.29262	13.1352	12.5928
512	1.10543	1.42676	25.7904	20.9864
768	1.37324	1.91994	60.7752	37.7672
1024	1.74783	2.65959	108.8208	41.9824
1280	2.24578	3.06519	165.58	46.2688
1536	2.33858	3.24225	243.208	54.78
2048	2.85203	3.94137	430.6584	71.656
2560	3.40617	5.01149	670.104	97.1848
3072	4.68989	6.00706	1015.5144	109.224
3584	4.3863	7.30431	1316.7928	135.6968
3968	5.64229	8.50355	1651.8271	139.9128

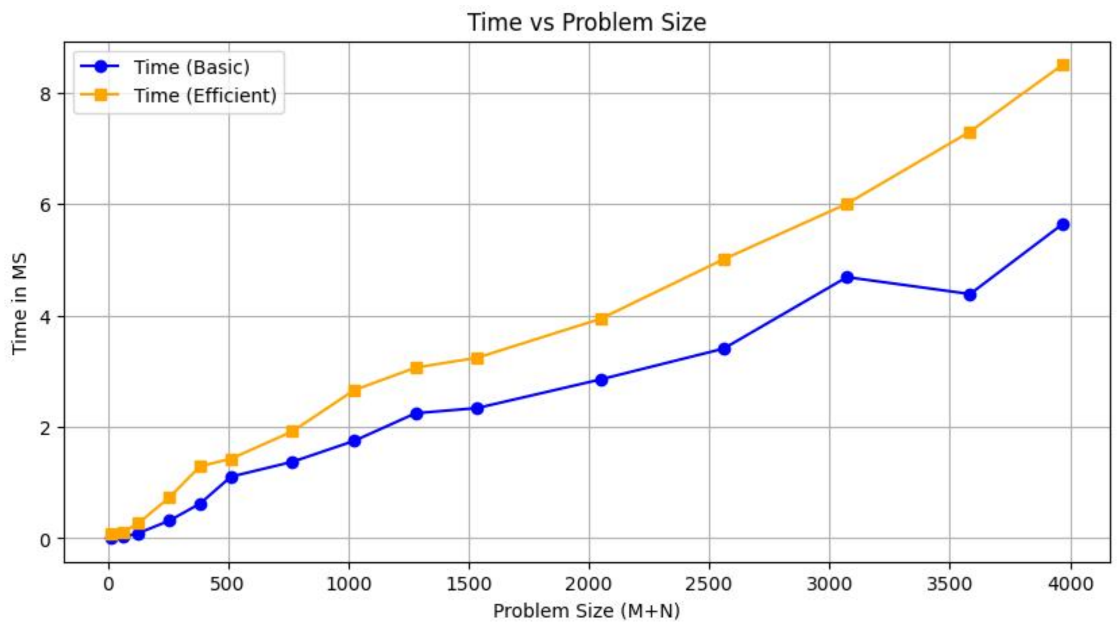
Insights

Graph1 – Memory vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)
Basic: Polynomial
Efficient: Linear
Explanation:
The Basic algorithm uses a dynamic programming approach that requires an m x n two-dimensional array dp to store intermediate results, where m and n are the lengths of the two input sequences, respectively. Therefore, the spatial complexity of this algorithm is O(mn), which is polynomial.
The Efficient algorithm uses a divide-and-conquer strategy to recursively divide the problem into subproblems. At each level of recursion, it only needs a one-dimensional array of length m or n to store the intermediate result. Therefore, at any given time, the space required for this algorithm is O(max(m, n)), which is linear.

Graph2 – Time vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)
Basic: Polynomial
Efficient: Ploynomial
Explanation:
Basic algorithm uses an algorithm based on dynamic programming, because m*n two-dimensional array needs to be filled in the algorithm, so its time complexity is O(mn), while Efficient algorithm uses a combination of divide and conquer dynamic programming, the two sequences are recursively divided into left and right halves. The cost of the optimal comparison results of the left and right halves is minimized, and the final time complexity should be O(2mn). However, the running time of the efficient algorithm is higher than that of the basic algorithm, but it is not twice as long, which may be due to the following reasons:
1. The current data size is not large enough, the running time is affected by some constant factors.

2. The recursive way of Efficient algorithms can also incur additional overhead.

Contribution

(Please mention what each member did if you think everyone in the group does not have an equal contribution, otherwise, write “Equal Contribution”)
9015551525 : <Equal Contribution>
4675328960 : <Equal Contribution>
8157970159 : <Equal Contribution>