SUMMARY

USC ID/s: 3684688813, 4391080611, 1825392034

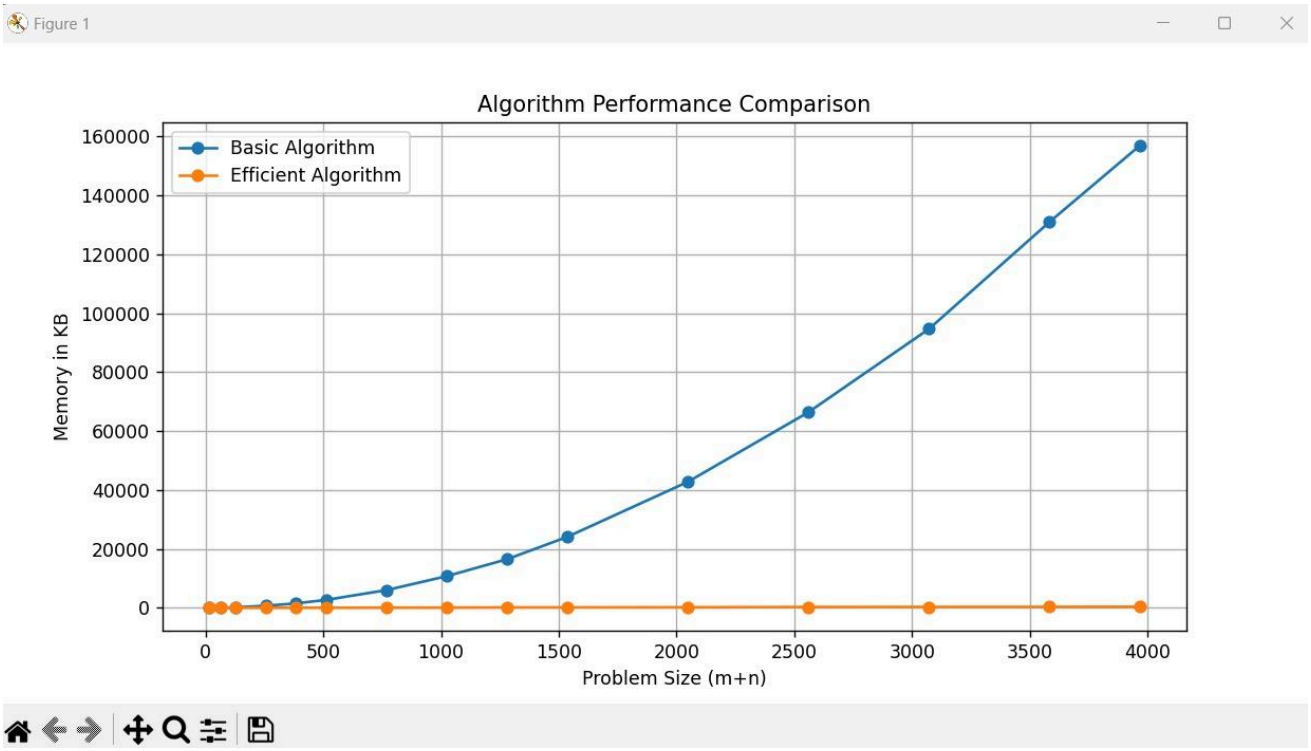
Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.0	0.0	8	4
64	1.0223388671875	3.0608177185058594	48	8
128	2.044200897216797	4.388570785522461	180	12
256	8.249282836914062	14.558315277099961	700	36
384	19.506216049194336	34.716606140136672	1512	48
512	32.590150833129888	58.518886566116211	2700	44
768	82.564830780029297	137.14957237243652	6012	72
1024	147.20630645751953	242.5692081451416	10748	80
1280	236.56058311462402	378.09085845947266	16492	160
1536	332.17597007751465	632.4350833892822	24088	164
2048	595.2494144439697	967.7562713623047	42764	196
2560	957.9994678497314	1536.0376834869385	66392	284
3072	1351.3827323913574	2174.5970249176025	94648	316
3584	1793.8203811645508	2927.746534347534	130968	332
3968	2312.0405673980713	3760.486125946045	156920	372

Insights

- In the context of the sequence alignment problem, the memory-efficient algorithm significantly reduces memory requirements from what is typically seen with traditional methods.
- If the sequences to be aligned are of length m and n the space complexity for these algorithms is $O(mn)$ which is polynomial.
- The memory-efficient algorithm, as discussed in class, alters this approach by reducing the space complexity to nearly linear.
- This is typically achieved by dividing the problem into smaller segments recursively.
- This method only needs to store two rows (or columns, depending on the implementation) of the matrix at any time, thereby reducing the space requirement to O(min(m, n)).

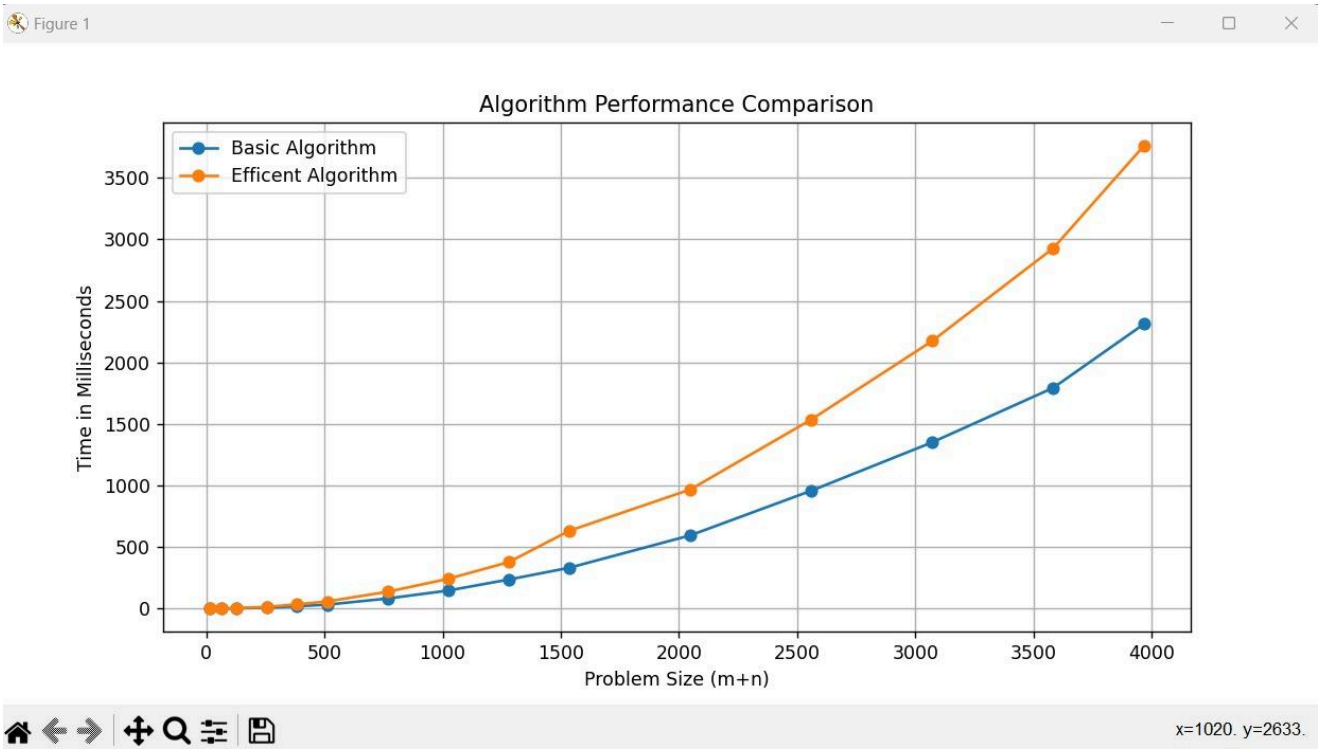
Graph1 – Memory vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)
Basic: Linear
Efficient: Polynomial

Explanation: The efficient algorithm significantly reduces memory usage by maintaining a nearly linear space requirement, compared to the basic algorithm's polynomial increase. This is achieved by using just two arrays, which allows for substantial memory savings and makes it feasible to process large datasets that the basic algorithm, with its need for a full two-dimensional matrix, would struggle to handle efficiently.

Graph2 – Time vs Problem Size (M+N)



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

Basic: Polynomial

Efficient: Polynomial

Explanation: The computational time for both algorithms increases polynomially, as depicted in the graph. The efficient algorithm, while performing twice as many operations as the basic algorithm, remains comparable in terms of its complexity. However, as the sum of m and n grows, the efficient algorithm progressively takes longer than the basic version of sequence alignment, as indicated by their respective curves.

Contribution

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

3684688813: Equal Contribution

4391080611: Equal Contribution

1825392034: Equal Contribution