SUMMARY

USC ID/s:

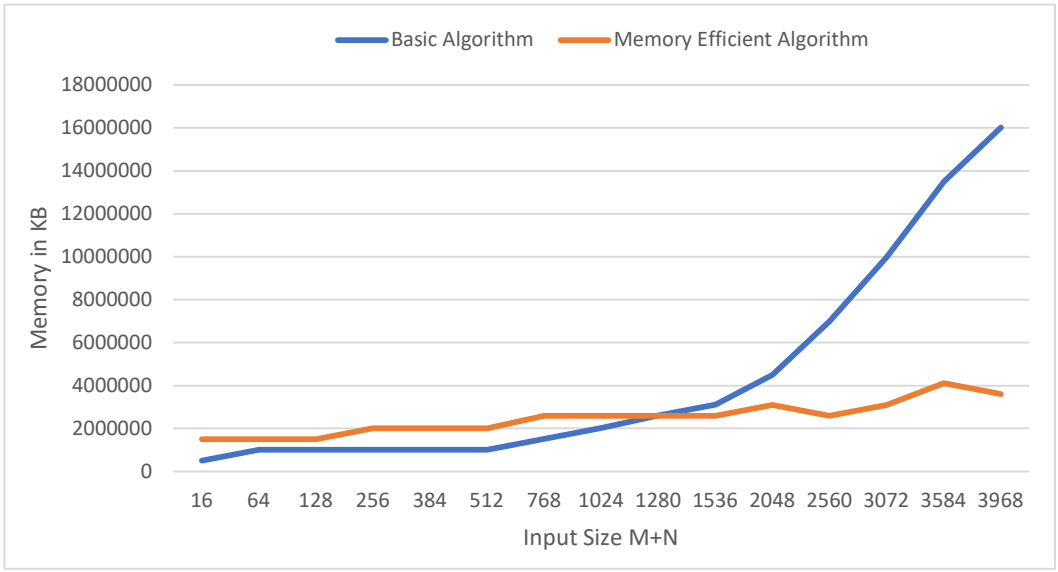
Zhuowen Fang 1628899825
Zirui Huang 9187076674
Zhengyang Wang 7114683586

Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.1127	5.4873	503368	1510016
64	0.3705	5.628	1006808	1510016
128	0.7167	7.1417	1006936	1510016
256	2.2998	10.9082	1007192	2013352
384	4.9928	13.6595	1007448	2013352
512	9.0964	14.9143	1007704	2013352
768	10.8735	18.0701	1513088	2600504
1024	12.9355	23.3231	2019496	2600488
1280	15.7199	28.9204	2603048	2600488
1536	17.1952	30.7012	3109968	2600496
2048	22.6307	42.3645	4500768	3103832
2560	26.9086	50.9769	6997448	2600488
3072	30.1724	52.3539	9975024	3103888
3584	35.736	67.3406	1.35E+07	4110624
3968	52.1373	73.744	1.60E+07	3607176

Insights

Graph1 – Memory vs Problem Size (M+N)



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

Basic: Polynomial

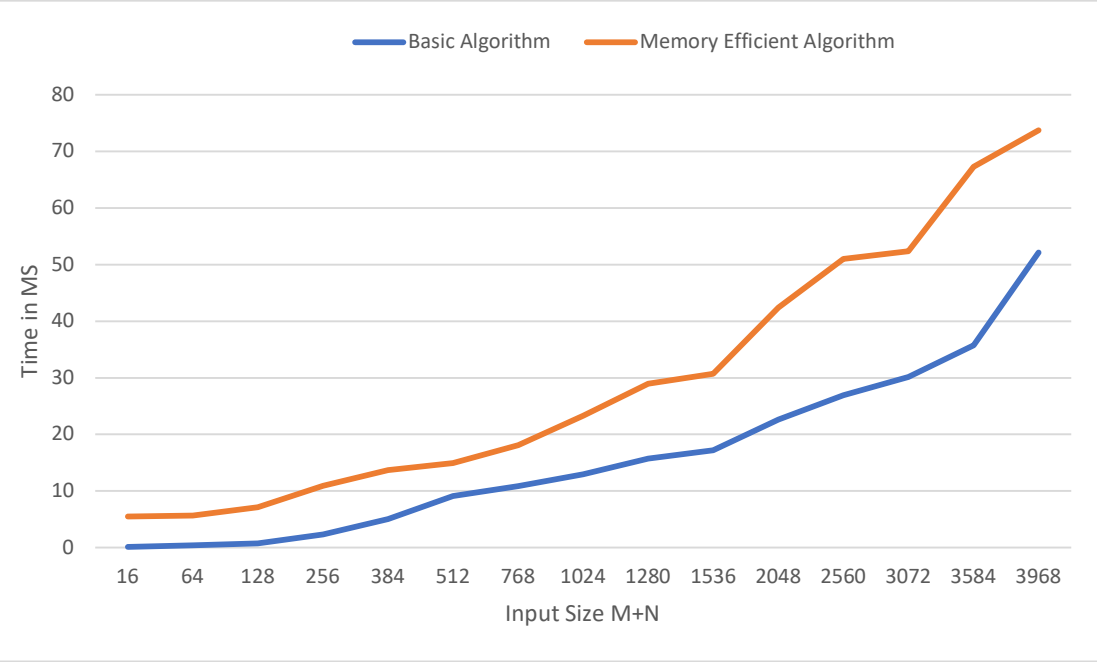
Efficient: Linear

Explanation:

The space complexity of the basic algorithm is O(MN) because we need to construct a complete dp table with M columns and N rows to keep track of the optimal results at each input size.

The space complexity of the efficient algorithm is O(M+N) because we are reusing a rolling column to keep track of the optimal results of the previous column.

Graph2 – Time vs Problem Size (M+N)



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

Basic: Polynomial

Efficient: Polynomial

Explanation:

The time complexity of both algorithms are O(MN). For the basic algorithm, we have to calculate the optimal result at each input size(Mi = 0 to M, Nj = 0 to N) to fill the dp table with size M X N. The calculation takes constant time at each step, so the resulting runtime is O(MN)

For the efficient algorithm, we use recusion to split the problem into smaller subproblems, and then combine the results. Adding the runtime of each level together we get a geometric sum equation, giving the total runtime of O(MN).

Contribution

<Equal Contribution>