SUMMARY

USC ID/s:

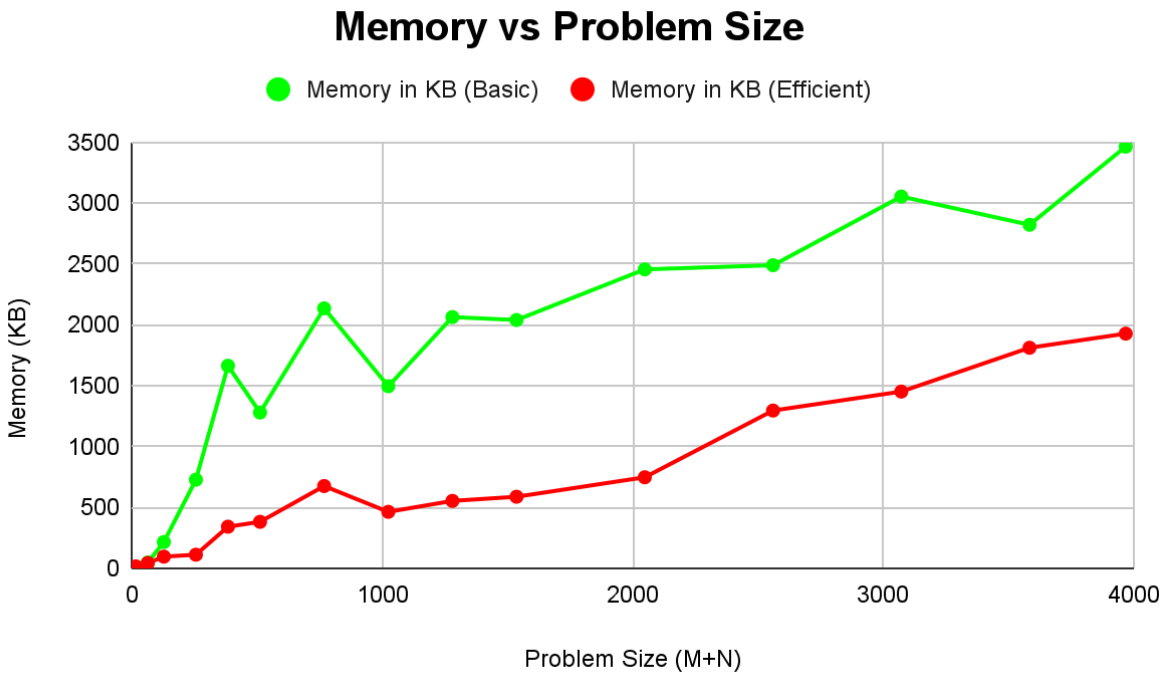
2963653580 (Nicholas Weber)
8940791833 (Joshua Maranan)

Datapoints

Input	Number of Tests	M+N	Average Time in MS (Basic)	Average Time in MS (Efficient)	Average Memory in KB (Basic)	Average Memory in KB (Efficient)
in1.txt	3	16	0.0	0.0	4	16
in2.txt	3	64	0.64	0.67	48	44
in3.txt	3	128	0.99	1.83	216	96
in4.txt	3	256	3.39	7.55	728	112
in5.txt	3	384	7.09	14.2	1664	342
in6.txt	3	512	12.67	24.98	1280	382
in7.txt	3	768	29.97	55.86	2136	676
in8.txt	3	1024	54.98	96.87	1496	464
in9.txt	3	1280	85.87	151.66	2064	554
in10.txt	3	1536	123.03	215.01	2040	588
in11.txt	3	2048	226.42	407.83	2456	748
in12.txt	3	2560	351.80	623.4	2490	1296
in13.txt	3	3072	512.73	887.39	3054	1452
in14.txt	3	3584	703.95	1232.90	2822	1812
in15.txt	3	3968	859.92	1527.98	3464	1928

Insights

Graph1 – Memory vs Problem Size (M+N)



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

Basic: Linear

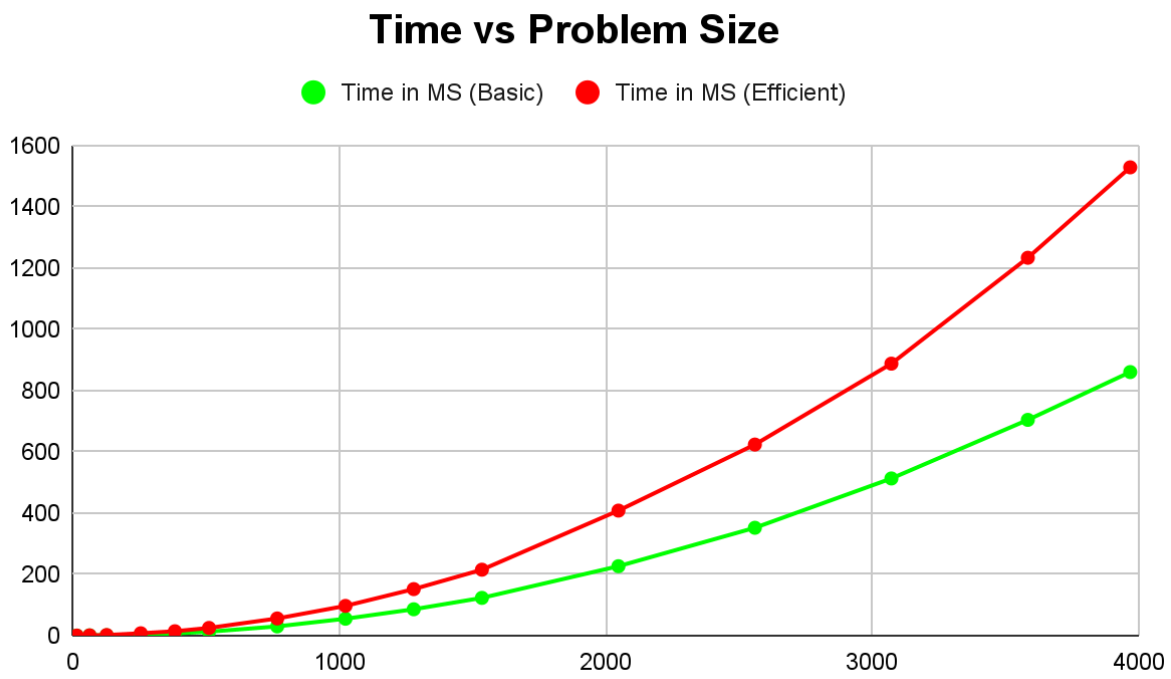
Efficient: Linear

Explanation:

In the graphed data, both the basic and efficient algorithms are increasing linearly with respect to the size of the input. However, we expected the basic algorithm to be of a polynomial nature because it requires O(N*M) memory to compute the dynamic programming table, where N and M are the lengths of the input strings. Behind the scenes this may be influenced and optimized by Python at runtime. Additionally for the efficient algorithm, the overhead cost of additional memory stacks for recursive function calls in Python is much higher than in some other languages, such as C++, which affects this data.

We see that the memory used from the basic algorithm consistently remains higher than that of the efficient algorithm. This is to be expected, as we discussed in class, we should see that basic would consume more memory but take less time than efficient.

Graph2 – Time vs Problem Size (M+N)



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

Basic: Polynomial

Efficient: Polynomial

Explanation:

Both the basic and efficient are increasing polynomially on the size of the input. Compared to the basic algorithm, the efficient algorithm increases in time much more than basic. As discussed in Lecture 8, the tradeoff for this was a decrease in memory used for the efficient algorithm, as we saw in Graph1.

Conclusion:

The data trends we see in the two graphs are in-line with the theoretical complexities we discussed in class, as the efficient algorithm uses less memory but more time than the basic algorithm.

Contribution

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

<USC ID/s>: <Equal Contribution>

2963653580: Equal Contribution

8940791833: Equal Contribution