SUMMARY

USC ID/s: 2725022497 (Sajan Gutta) → only member, this was done individually

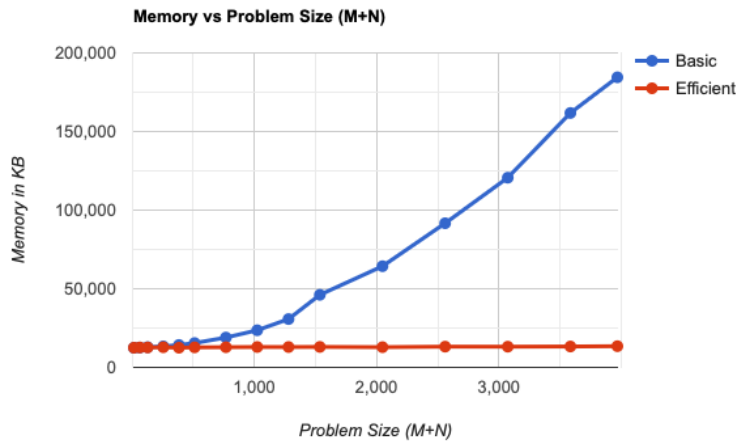
Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.484	0.490	12456	12512
64	1.025	2.497	12600	12656
128	3.721	6.057	12836	12668
256	10.885	18.640	13368	12812
384	23.073	39.345	14248	12496
512	40.118	71.477	15472	12740
768	93.636	158.110	18984	12840
1024	176.115	279.106	23612	12964
1280	282.429	464.368	30736	12944
1536	403.196	647.626	46148	13020
2048	809.953	1164.641	64352	12860
2560	1150.130	1838.610	91672	13192
3072	1647.661	2573.361	120644	13168
3584	2328.284	3596.280	161800	13288
3968	2821.242	4426.554	184492	13484

INSIGHTS CONTINUE ON NEXT PAGE

Insights

Graph1 – Memory vs Problem Size (M+N)



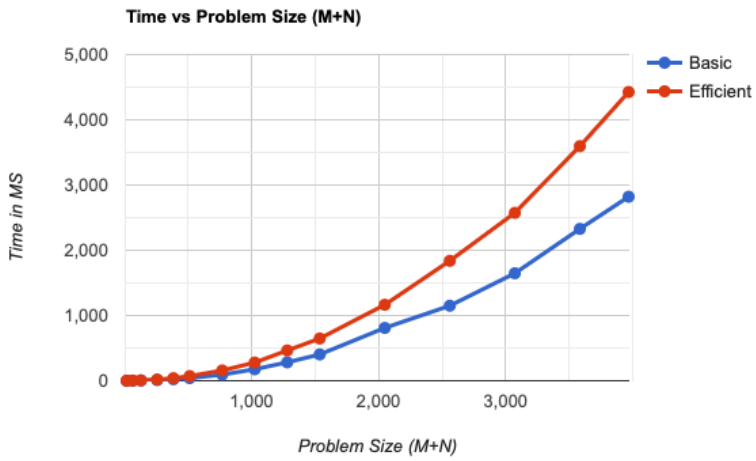
Nature of the Graph (Logarithmic/ Linear/ Exponential)
Basic: Exponential
Efficient: Linear

Explanation: We discussed this difference in class. The basic solution grows extremely fast in memory usage as the memory depends on M*N. For example, if M and N are 3 billion each, we need a 3 billion by 3 billion array for calculating opt values under the basic solution. This growth depends on both M and N multiplied which generates an exponentially growing memory usage for the solution.

Conversely, the efficient solution only needs 2 columns of length M – where M is the first string – for calculating the opt values in an array. Thus, it really grows more linearly with respect to the problem size and is much more efficient. Above, the line for the efficient solution doesn't reflect much growth at all because there is a base memory usage of about 12,000 KB that gets used regardless of problem size. This memory is associated with things like reading in file input, storing/generating the strings to align, etc. However, the subsequent additional memory needed as problem size grows see linear growth and grows very slowly. By the time we get to big problem sizes, the efficient solution's memory savings begin to make a significant difference.

See Time vs. Problem Size on Next Page

Graph2 – Time vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Exponential)

Basic: Linear
Efficient: Linear

Explanation: As we discussed in class, the time complexity of the basic implementation is O(mn) while the Efficient implementation is O(2cmn) which is also O(mn). Basically, both growth linearly with respect to the problem size. In this case, the problem size consists of M and N, both of which reflect its actual size as they are the size of the input strings. Thus, both of these graphs depict efficient solutions that are actually polynomial time.

The "efficient" solution is meant to be more memory efficient as previously discussed. However, it has the same time complexity as the basic implementation, albeit with a higher constant as you can see by it typically having a higher value for time in the graph above. We also discussed this during lecture.

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>

2725022497: All Contribution (project was done individually)