SUMMARY

USC ID/s:

3281934045

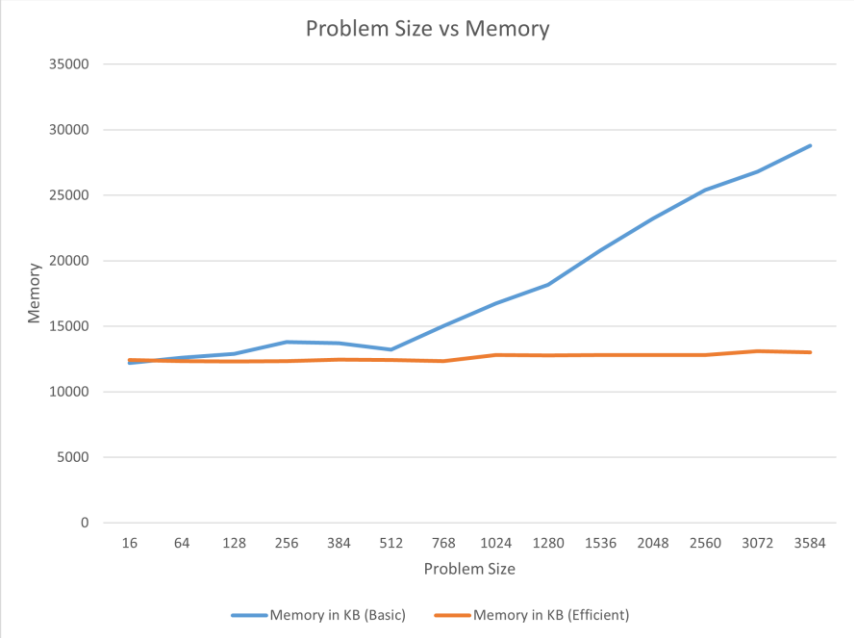
Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.10	0.19	12204	12436
64	0.61	1.65	12192	12436
128	3.10	5.45	12620	12352
256	8.17	17.79	12908	12312
384	18.24	36.72	13808	12344
512	42.46	65.51	13716	12448
768	91.98	145.27	13220	12420
1024	177.36	262.05	15024	12352
1280	237.65	432.11	16748	12824
1536	364.31	604.14	18188	12780
2048	708.78	1084.37	20796	12808
2560	1010.89	1690.40	23216	12812
3072	1519.85	2521.86	25392	12812
3584	2021.03	3227.96	26796	13108
3968	2606.02	4108.90	28792	13024

Insights

In this project, the algorithms were implemented in Python3. Programs were executed on Windows, Mac and Ubuntu. Result data from all platforms demonstrate the same conclusion that the efficient version uses much less memory but requires more running time than the basic version. However, the data have significant difference across these 3 platforms. Eventually, data from Ubuntu was picked for this report.

Graph1 – Memory vs Problem Size (M+N)



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

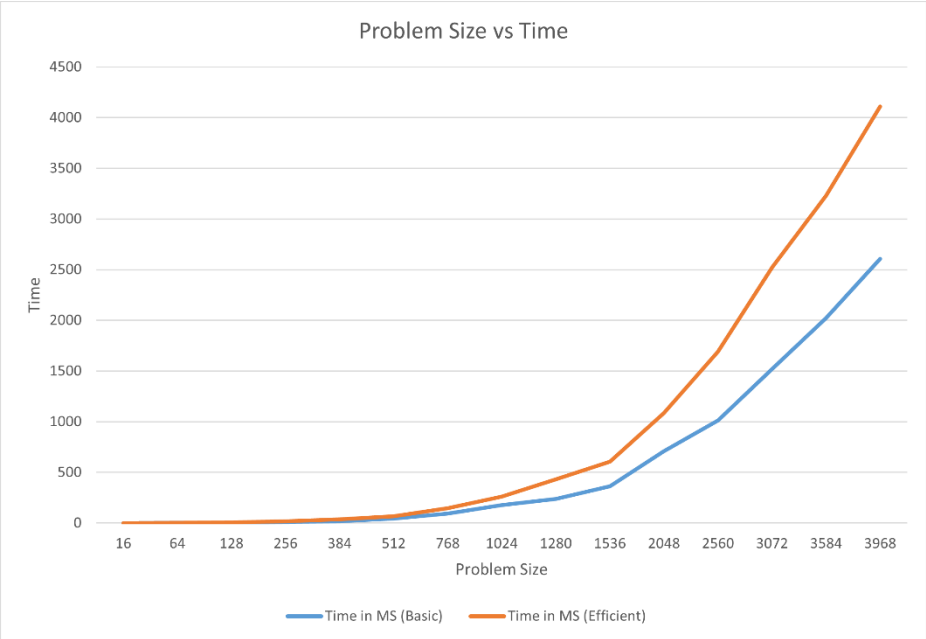
Basic: logarithmic

Efficient: polynomial

Explanation:

As size grows, the line of basic version grows in polynomial speed, while the line of efficient version is almost a horizontal line. Apparently, the efficient version uses much less memory when the size is huge.

Graph2 – Time vs Problem Size (M+N)



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

Basic: polynomial

Efficient: polynomial

Explanation:

Both lines grow in polynomial speed as problem size increases, but the efficient version costs more time.

Contribution

3281934045