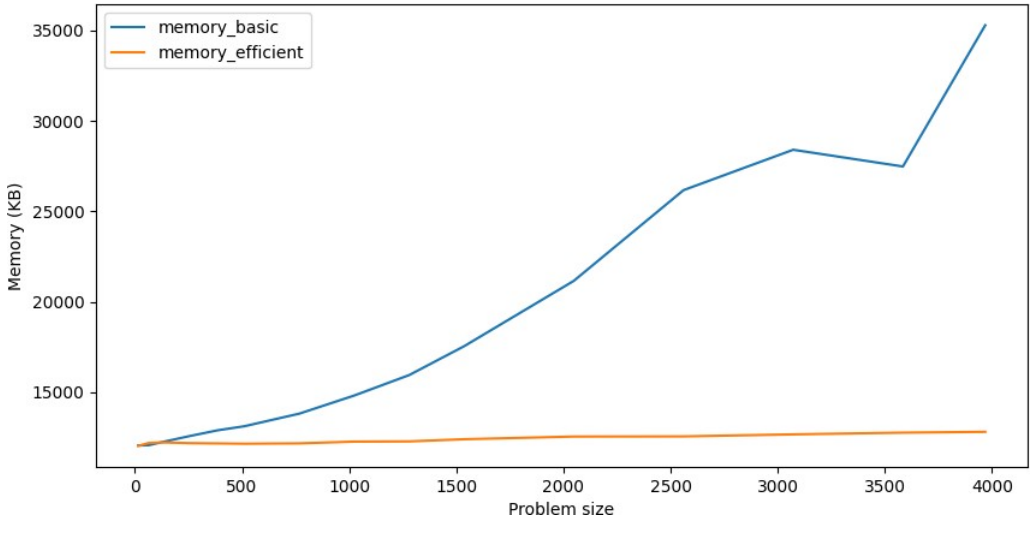
SUMMARY

Datapoints

M+N	Time in MS (Basic)	Time in MS (Efficient)	Memory in KB (Basic)	Memory in KB (Efficient)
16	0.1010894775390625	0.2532005310058594	12068	12044
64	1.1680126190185547	2.374887466430664	12080	12204
128	4.917144775390625	8.749246597290039	12256	12236
256	16.791105270385742	32.180070877075195	12588	12196
384	43.66302490234375	74.43094253540039	12900	12180
512	73.16184043884277	126.75213813781738	13132	12164
768	166.08715057373047	297.03497886657715	13828	12184
1024	301.03015899658203	526.4461040496826	14828	12280
1280	467.47398376464844	809.9651336669922	15956	12292
1536	690.2799606323242	1121.4079856872559	17548	12412
2048	1234.666109085083	2066.4689540863037	21160	12560
2560	1901.0639190673828	3270.8218097686768	26172	12568
3072	2841.459274291992	4501.637935638428	28404	12680
3584	3855.5550575256348	6274.60503578186	27480	12268
3968	4584.528207778931	7847.738742828369	35284	12820

Insights

Graph1 – Memory vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Exponential)

Basic: Exponential

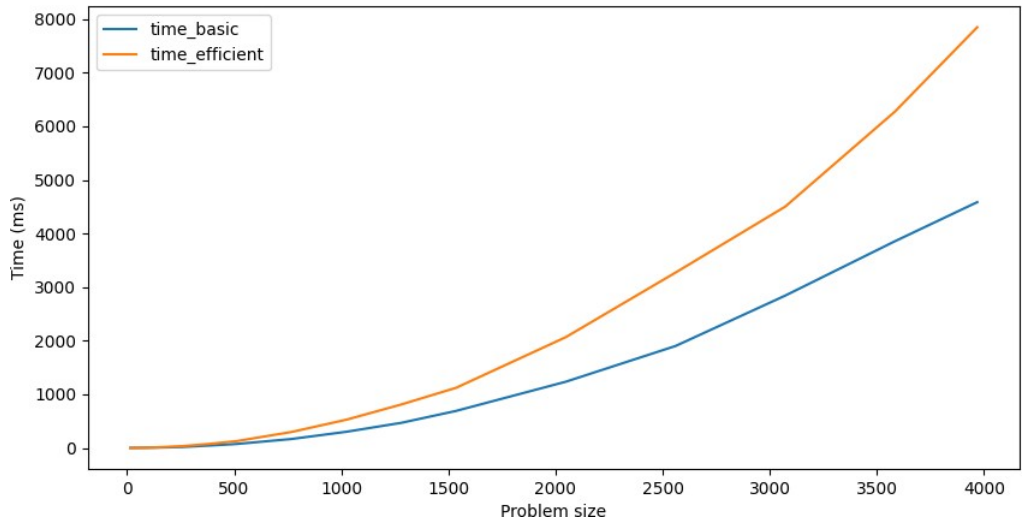
Efficient: Linear

Explanation:

The memory efficient algorithm decreases the space requirement, by incorporating the divide and conquer strategy into the dynamic programming process. Thus, we observed that the memory efficient technique was considerably more efficient than the basic algorithm.

The memory efficient algorithm plot displayed almost linear trends in memory growth as opposed to exponential memory growth of basic algorithm. The difference in memory usage was more noticeable as the problem size increases.

Graph2 - Time vs Problem Size (M+N)



Nature of the Graph (Logarithmic/ Linear/ Exponential)

Basic: Quadratic

Efficient: Quadratic

Explanation:

Both basic and memory efficient algorithms have similar quadratic trends in plot, however on observing the datapoint values we noticed the memory efficient algorithm takes approximately twice as long to run as basic algorithm. This becomes more apparent when the problem sizes are larger.