**SUMMARY**

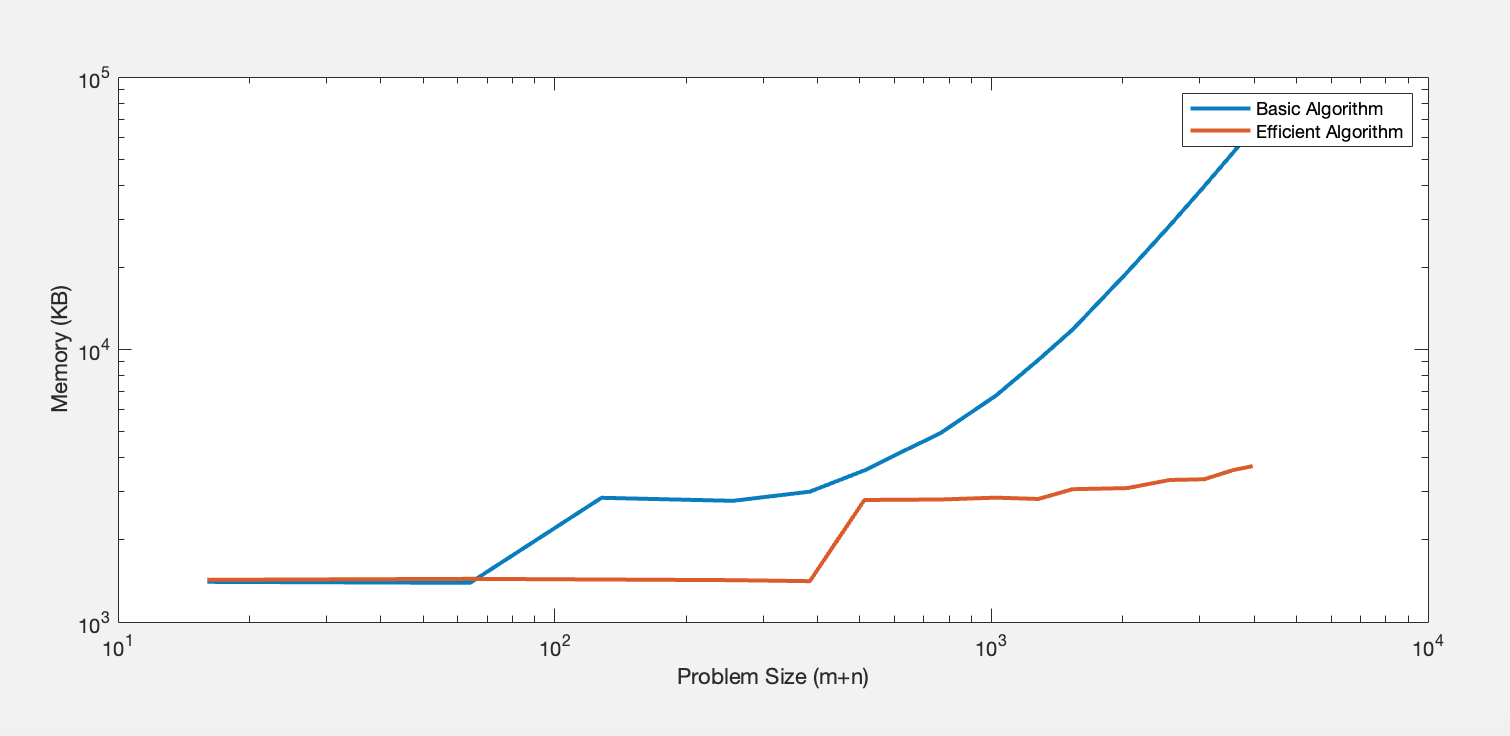
## USC ID/s: 1417356257(Qingyu Zhou) ; 5804740637(Rajasimha Reddy Jerry Sivaram Reddy)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS (Basic) | Time in MS (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 0.15 | 0.255 | 1404 | 1428 |
| 64 | 0.393 | 0.187 | 1392 | 1440 |
| 128 | 0.625 | 0.561 | 2856 | 1432 |
| 256 | 0.602 | 0.505 | 2784 | 1424 |
| 384 | 1.719 | 1.084 | 3008 | 1412 |
| 512 | 2.135 | 1.997 | 3600 | 2804 |
| 768 | 3.647 | 3.37 | 4956 | 2816 |
| 1024 | 5.099 | 6.97 | 6768 | 2860 |
| 1280 | 8.985 | 9.133 | 9144 | 2828 |
| 1536 | 10.588 | 10.58 | 11840 | 3072 |
| 2048 | 18.471 | 24.314 | 19208 | 3100 |
| 2560 | 28.631 | 34.672 | 28452 | 3320 |
| 3072 | 42.437 | 45.186 | 39664 | 3340 |
| 3584 | 54.63 | 68.499 | 52968 | 3612 |
| 3968 | 73.002 | 104.901 | 64356 | 3732 |

## Datapoints

## Insights

### Graph1 – Memory vs Problem Size (M+N)



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

Basic: Polynomial

Efficient: Linear

#### Explanation: The basic algorithm builds a (m+n+1)^2 array storing the minimum costs, so the memory used by basic algorithm is O(n^2). The efficient algorithm only takes a 2\*(m+1) array for minimum costs, which is O(n).

### Graph2 – Time vs Problem Size (M+N)

Chart, line chart

Description automatically generated

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

Basic: Polynomial

Efficient: Polynomial

#### Explanation: Both of the basic algorithm and efficient algorithm need to calculate (n+1)\*(m+1) elements of the minimum cost table, so the time of both algorithms is O(n^2).

## 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>

1417356257 : Equal Contribution

5804740637 : Equal Contribution