**SUMMARY**

## USC ID/s:

## 7418934031

## 8519965458

## 9347096813

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS (Basic) | Time in MS (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 1.0001659393310547 | 0.7100105285644531 | 2.0 | 16 |
| 64 | 1.9958019256591797 | 1.9931793212890625 | 88.0 | 24 |
| 128 | 4.986286163330078 | 8.975505828857422 | 156.0 | 32 |
| 256 | 12.996196746826172 | 27.918100357055664 | 632.0 | 96.0 |
| 384 | 33.90836715698242 | 75.30951499938965 | 1488.0 | 96.0 |
| 512 | 66.83683395385742 | 103.72161865234375 | 2624.0 | 112.0 |
| 768 | 132.645845413208 | 247.34115600585938 | 6008.0 | 112.0 |
| 1024 | 241.10746383666992 | 470.7345962524414 | 10576.0 | 92.0 |
| 1280 | 351.0739803314209 | 696.2287425994873 | 16528.0 | 168.0 |
| 1536 | 506.64472579956055 | 1041.0325527191162 | 23516.0 | 320.0 |
| 2048 | 955.1734924316406 | 2218.8665866851807 | 41808.0 | 360.0 |
| 2560 | 1464.069128036499 | 3472.6901054382324 | 65392.0 | 504.0 |
| 3072 | 2049.670457839966 | 4580.138683319092 | 93928.0 | 536.0 |
| 3584 | 2985.949993133545 | 6155.249357223511 | 127840.0 | 800.0 |
| 3968 | 3555.0918579101562 | 7175.60887336731 | 156604.0 | 736.0 |

## Datapoints

## Insights

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

图表, 折线图

描述已自动生成

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

Basic: Polynomial

Efficient: Linear

#### Explanation:

The basis Dynamic programming algorithm create a M\*N table space to calculate the optimal cost and find the minimum cost by backtracking. So it takes O(M\*N), as M+N growth, it increase in polynomial.

The efficient algorithm recursively divide string A in half and use a total 2\*len(stringB) memory space to calculate the cost of dynamic programming for every divided parts. The actual memory cost for this algorithm is a 2\*N table space to the DP process. So it is O(N) or O(M)

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

图表, 折线图

描述已自动生成

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

Basic: Polynomial

Efficient: Polynomial

#### Explanation:

Both time costs of the two algorithms is Polynomial.

The time cost for basic algorithm is O(M\*N) as it calculates the M\*N DP table.

The time cost for efficient algorithm is O(M\*N) as well. It is Roughly 2\* what the basic algorithm takes. As each time we divide our strings into parts, we calculate the DP space of 1\*M\*N+(1/2)\*M\*N+(1/4)\*M\*N+…… ≈2\*M\*N

## Contribution

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

Equal Contribution