**SUMMARY**

## USC ID/s:

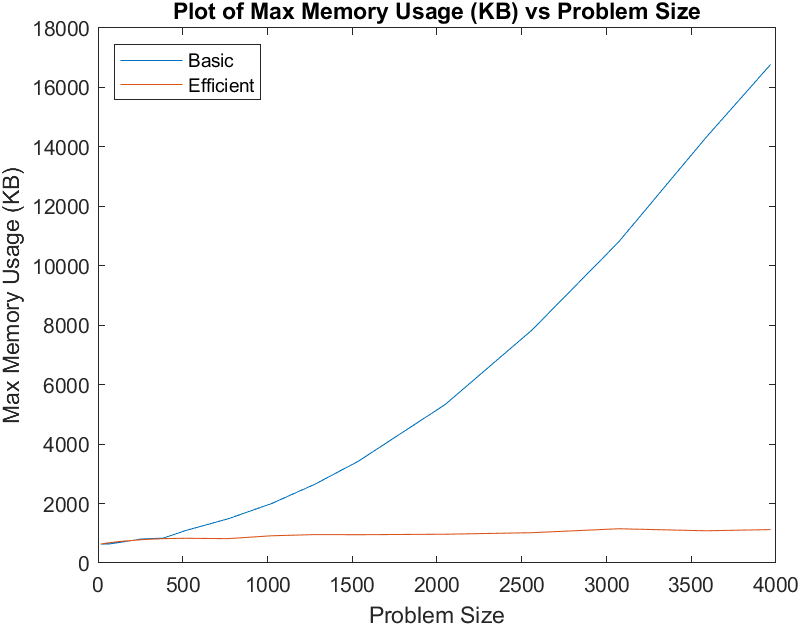
9976440909

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS (Basic) | Time in MS (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 0.148 | 0.17 | 636 | 640 |
| 64 | 0.67 | 1.276 | 640 | 680 |
| 128 | 2.149 | 4.035 | 692 | 724 |
| 256 | 7.365 | 13.781 | 808 | 784 |
| 384 | 18.157 | 31.097 | 836 | 820 |
| 512 | 26.397 | 47.512 | 1084 | 832 |
| 768 | 64.725 | 119.988 | 1488 | 820 |
| 1024 | 119.103 | 220.16 | 2000 | 916 |
| 1280 | 157.136 | 305.637 | 2652 | 956 |
| 1536 | 151.143 | 289.596 | 3424 | 952 |
| 2048 | 376.547 | 750.159 | 5332 | 968 |
| 2560 | 630.382 | 1212.42 | 7836 | 1020 |
| 3072 | 803.78 | 1551.7 | 10804 | 1152 |
| 3584 | 1049.88 | 2053.37 | 14296 | 1084 |
| 3968 | 1416.22 | 2806.62 | 16772 | 1124 |

## Datapoints

## Insights

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



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

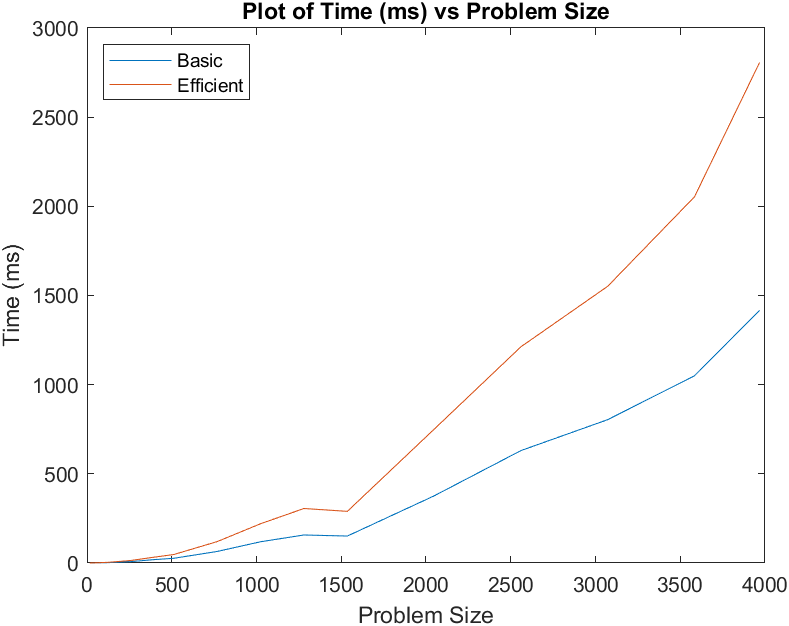
Basic: Exponential

Efficient: Logarithmic or Linear

#### Explanation:

The basic graph is best modelled by an exponential function. As the problem size increases, we see the basic graph’s slope continues to increase, which results in an exponential form with respect to problem size. For the efficient graph, we see the graph shoot up quickly while problem sizes are near 0 but begin to level out (with some variation) as the problem size increases. This is somewhat expected: the base program requires a certain amount of memory, including string building and file parsing, which is shared between the basic and efficient algorithms. Overall, this lends itself to follow a logarithmic form, although there appear to be specific problems that do not follow a perfect curve – as a result, the graph could also be modelled by a linear function, if we smooth out the noise/variation present in the data points on the efficient dataset. In the above graph, the sharp increase in memory usage near problem size 0 for the efficient algorithm is not noticeable due to the vast size difference between the basic and efficient programs.

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



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

Basic: Exponential

Efficient: Exponential

#### Explanation:

As problem size increases, the number of calculations we need to do increases exponentially. Each additional character included in our string alignment results in 3 options: a blank paired with a character, two matched characters, or two mismatched characters. As a result, each possibility branches off and requires its own set of calculations until the end of the strings are reached. This fundamental logic is unchanged between the two algorithm variations: we see that the memory efficient algorithm tends to take double the time because it requires double the computations to come to a string alignment, as expected. As problem size increases, we see the slope of each graph continue to increase overall, which makes these two graphs best modelled by an exponential function with respect to the problem size.

## 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>