# Text Description automatically generated with medium confidence

# Ninad Bharat Gund

# ASU ID: 1222336947

# Experiments:

# Tests of compression ratios by testing different inputs –

# anne-of-avonlea

# n: 491713

# sum: 2339424

# numberOfBits: 4.7577

# fixedNumberOfBits: 7

# compressionRatio: 32.0328 %

# haiku1

# n: 62

# sum: 275

# numberOfBits: 4.43548

# fixedNumberOfBits: 7

# compressionRatio: 36.6359 %

# she-sells

# n: 39

# sum: 140

# numberOfBits: 3.58974

# fixedNumberOfBits: 7

# compressionRatio: 48.7179 %

# tale-of-two-cities ch1

# n: 7425

# sum: 35679

# numberOfBits: 4.80525

# fixedNumberOfBits: 7

# compressionRatio: 31.3535 %

# tongue-twisters

# n: 374

# sum: 1790

# numberOfBits: 4.7861

# fixedNumberOfBits: 7

# compressionRatio: 31.6272 %

# 

# Test of running time for both algorithms, tested as a function of the input size (n) –

# (The result has some outliers because test data was created at random, based on the instance, algorithms can have best/worth running times. Insertion sort’s running time can be very different for a relatively sorted vs reverse ordered input. So we will choose to focus on the trend of the plot instead)

# As expected, insertion sort has a order of growth of O(n2), while merge sort grows at O(n\*logn)

# For a very small input size (below 200 in this case), insertion sort performs better, but as the input size grows, merge sort becomes significantly faster. So we can state that insertion sort can be better for really small input instances, while merge sort is better for large input instances.

# Test of encoding/decoding time as a function of input size (n) –

# Chart, line chart Description automatically generated

# Test of compression ratio as a function of input size (n) –

# Chart, line chart Description automatically generated

# It can be observed that the compression ratio becomes worse as the size of the input increases, but this is only true for change in input size for lower range. After a point (374+ in this case), the compression ratio is barely affected by the size of the input.

# The behavior is likely because in small inputs, not all the symbols are present and thus we do not encode the symbols with 0 occurrences. After the input size grows significantly, almost all the symbols are present, and we must encode a greater number of symbols. So, we can say that compression ratio is more of a function of the number of symbols to be encoded and not the size of the input directly.