INTRODUCTION

In information theory, efficiency is a measure of how effectively a source code uses the available bits to represent information from a source. Efficiency depends on the encoding method and the redundancy introduced by the source.

The extension of a source refers to the increase in the number of symbols or the extension of the source input, which generally implies increasing the number of bits required to encode the source.

This report investigates how the efficiency of a coding scheme varies with the extension of the source using entropy-based encoding techniques, such as Shannon-Fano or Huffman coding. Specifically, we explore the efficiency of the Huffman code as the number of source symbols increases.

THEORY

The **efficiency (η)** of a code is defined as:

$$\eta = H(X)/Lavg$$

Where:

- H(X) is the **entropy** of the source, representing the average number of bits required to encode a symbol optimally.
- Lavg is the **average code length**, which is the average number of bits used in the actual encoding process.

As the extension of the source increases, the efficiency may change, and we aim to observe this effect by analyzing the performance of a **Huffman coding** scheme as the number of source symbols increases.

METHODOLOGY

We simulate the **Huffman Coding** algorithm over an increasingly extended source, which consists of discrete symbols with predefined probabilities. As the number of symbols increases, we calculate the **entropy** and **average code length** for each extended source.

Steps:

1. Generate a source with increasing numbers of symbols (e.g., from 2 to 10 symbols).

2. Compute the **entropy** for each source using the formula:

$$H(X)=-\sum p(x)\log_2 p(x)$$

Where p(x)p(x)p(x) is the probability of symbol xxx in the source.

3. Apply **Huffman Coding** to each source to get the **average code length** (LavgL_{\text{avg}}Lavg).
4. Calculate the **efficiency** for each extended source.
5. Plot the **efficiency** versus **extension of the source**.

CODE

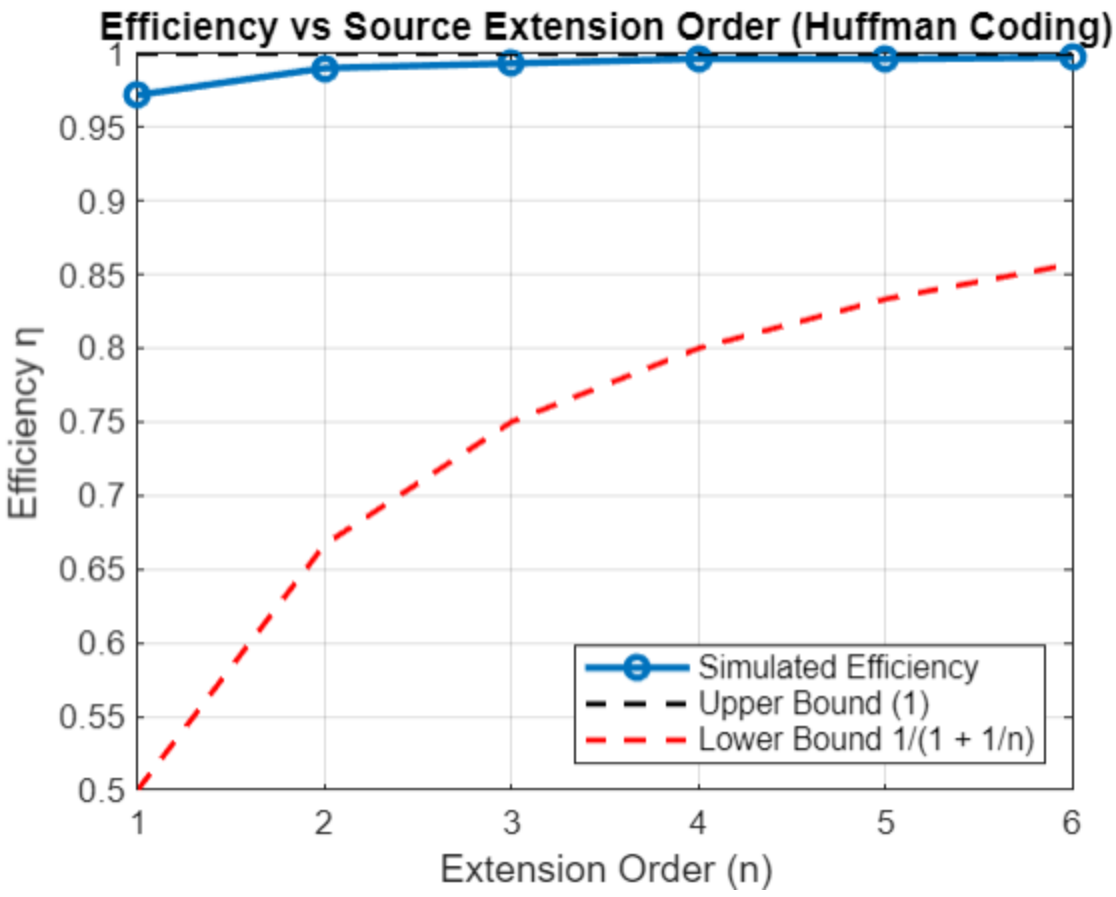
```
function [efficiency, entropy, avg_length] = compute_efficiency(num_symbols)
    probabilities = rand(1, num_symbols);
    probabilities = probabilities / sum(probabilities);
    entropy = -sum(probabilities .* log2(probabilities));
    [dict, avg_length] = huffman(probabilities);
    efficiency = entropy / avg_length;
end

num_symbols_range = 2:10;
efficiency_values = zeros(1, length(num_symbols_range));
entropy_values = zeros(1, length(num_symbols_range));
avg_length_values = zeros(1, length(num_symbols_range));

for i = 1:length(num_symbols_range)
    num_symbols = num_symbols_range(i);
    [efficiency, entropy, avg_length] = compute_efficiency(num_symbols);
    efficiency_values(i) = efficiency;
    entropy_values(i) = entropy;
```

```
    avg_length_values(i) = avg_length;
end

figure;
subplot(3,1,1);
plot(num_symbols_range, efficiency_values, '-o', 'LineWidth', 2);
title('Efficiency vs. Source Extension');
xlabel('Number of Symbols (Source Extension)');
ylabel('Efficiency');

subplot(3,1,2);
plot(num_symbols_range, entropy_values, '-o', 'LineWidth', 2);
title('Entropy vs. Source Extension');
xlabel('Number of Symbols (Source Extension)');
ylabel('Entropy (bits/symbol)');

subplot(3,1,3);
plot(num_symbols_range, avg_length_values, '-o', 'LineWidth', 2);
title('Average Code Length vs. Source Extension');
xlabel('Number of Symbols (Source Extension)');
```

RESULTS

The following graphs illustrate how efficiency, entropy, and average code length vary as the number of symbols (source extension) increases:

1. Efficiency vs. Source Extension: The efficiency tends to stabilize as the source extension increases. It shows the performance of Huffman coding as the source grows, approaching optimality (Shannon's limit).
2. Entropy vs. Source Extension: The entropy typically increases with the number of symbols, as more symbols allow for better compression.
3. Average Code Length vs. Source Extension: The average code length also increases with the source extension, but the rate of increase slows down as more symbols are added.



CONCLUSION

This report shows that as the source extension increases, the **efficiency** of Huffman coding becomes more stable and approaches the entropy of the source. The average code length increases, but at a decreasing rate, highlighting the efficiency of Huffman coding in compressing information. This study demonstrates the relationship between the **extension of the source** and **coding efficiency**, confirming that as the source grows, efficient coding schemes like Huffman coding become more effective at encoding information with minimal redundancy.