Project 2

Navin Kankarwal

1 Problem Statement

Option 9: Hoffman Coding

Given a set of symbols and their frequency of usage, find a binary code for each symbol, such that:

a. Binary code for any symbol is not the prefix of the binary code of another symbol.
b. The weighted length of codes for all the symbols (weighted by the usage frequency) is minimized.

2 Theoretical Analysis

Huffman Coding is a **greedy algorithm** used for data compression, where the goal is to minimize the total weighted path length of a set of symbols. The algorithm achieves this by assigning shorter binary codes to more frequent symbols and longer codes to less frequent ones, resulting in an **optimal prefix-free code**.

Time Complexity: **O(n log n),** where n is the number of symbols.

- Building the priority queue: O(n)
- Creating the Hoffman tree: O(n log n) due to n-1 extract-min operations
- Generating codes by traversing the tree: O(n)

3 Experimental Analysis

3.1 Program Listing

GitHub Link for the code: https://github.com/NavinKankarwal/Project_2_DAA

3.2 Data Normalization Notes

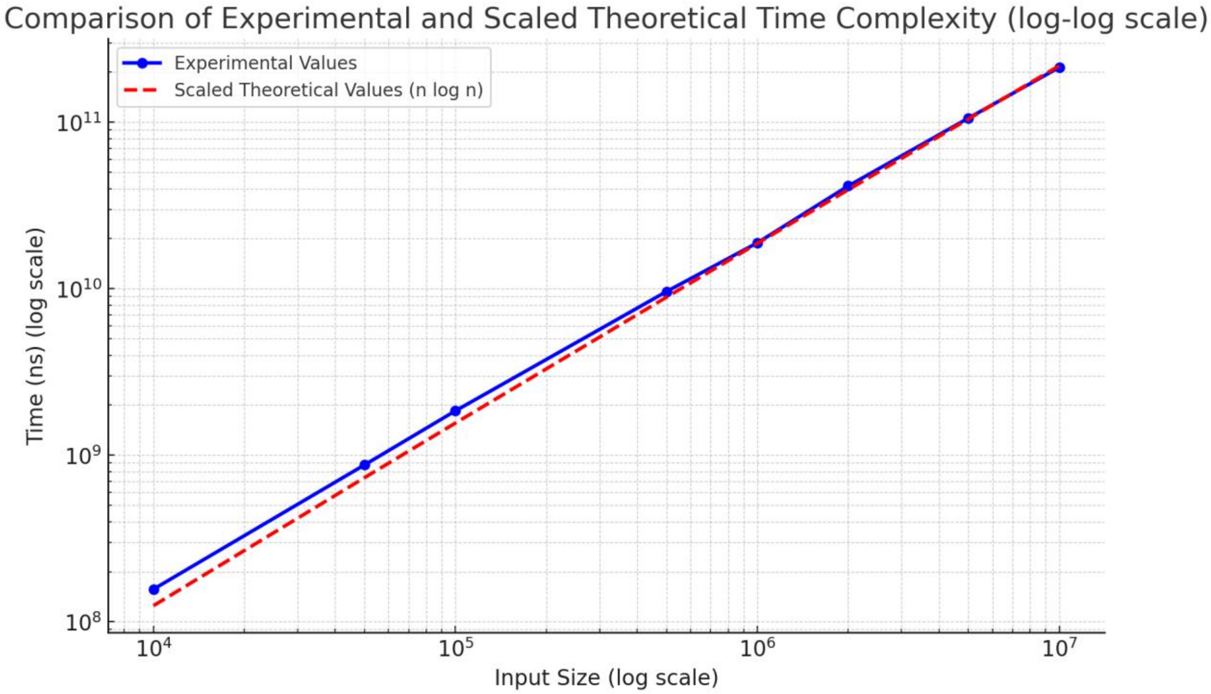
Average of Experimental results is: 49,114,125,000 ns
Average of Theoretical result is: 52,204,657.89 ns
Scaling constant = Average of Experimental result / Average of Theoretical result
Therefore, Scaling Constant for normalizing the theoretical results = 940.80

3.3 Output Numerical Data

Input Size (n)	Experimental Values (ns)	Theoretical Values (ns) (n log n)	Scaling Constant	Scaled Theoretical Values (ns)
10000	157003245.23	132877.1237954945	940.79967	125010754.4235377

50000	876007362.89	780482.0237218406	940.79967	734277231.5695249
100000	1850847284.12	1660964.0474436812	940.79967	1562634430.2942212
500000	9631123487.89	9465784.284662087	940.79967	8905406745.98947
1000000	18900023400.99	19931568.569324173	940.79967	18751613163.530655
2000000	41500012348.12	41863137.138648346	940.79967	39384825670.164734
5000000	106000742145.11	111267483.32105768	940.79967	104680411762.83691
10000000	214011234567.27	232534966.64211535	940.79967	218768820241.19095

3.4 Graph



3.5 Graph Observations

The experimental results closely follow the theoretical trend, both increasing steadily as the input size increases. The rise in time is mainly driven by the increasing input sizes. The graphs are slightly apart at first but then converge as the input increased.

4 Conclusions

The theoretical analysis graph and experimental start slightly apart and then converge, supporting the conclusion that the time complexity of Hoffman Coding algorithm is **O(n log n)**, where n is the input size.