**Internship report**

**on**

**Design** **of Encoding – Decoding Algorithm and**

**Implementation using Web Development and Data Structures**

*by*

**KUNAL MEHTA (2018UGEC016)**

*Under the supervision of*

**Dr. Prashant Kumar**

(Assistant Professor, Department of ECE, NIT Jamshedpur)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

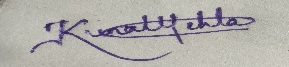
**NATIONAL INSTITUTE OF TECHNOLOGY JAMSHEDPUR**

**December 2021**

**CANDIDATE’S DECLARATION**

We hereby declare that

1. I have followed all the guidelines provided by the Institute in preparing the report.
2. I have conformed to the norms and guideline given in the Ethical Code of Conduct of my Institute.
3. Wherever I have used materials (data, theoretical analysis, figures and texts) from other sources, I have given due credit to them by citing them in the project report and giving their details in the reference. Further I have taken permission from the copyright owners of the sources, wherever necessary.



Signature of the Student

Kunal Mehta

(2018UGEC016)

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**National Institute of Technology Jamshedpur, Jamshedpur**



**CERTIFICATE**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

This is to certify that the Internship report entitled, “**Design of Encoding – Decoding Algorithm and Implementation using Web Development and Data Structures**” submitted by “***Kunal Mehta (2018UGEC016)***” B.Tech (Hons.) students of ECE department of NIT Jamshedpur, India is a record of bona-fide Work carried out by them under my supervision. This Internship report is submitted in partial fulfilment for the requirement of Industrial training in B,Tech degree. The work incorporated in this internship report has not been, to the best of my knowledge, submitted to any other university or Institute for the award or any degree or diploma.

**Dr. Prashant Kumar**

(Internship Supervisor)

Examined and Approved

The viva-voice exam has been held on -------------------- in online mode at WebEx platform.

**Faculty Advisor** **Head of Department (ECE)**

**ACKNOWLEDGEMENT**

Firstly, I would like to express my gratitude and sincere thanks to **Assistant Prof. Dr. Prashant Kumar Sir**, my intern supervisor, Department of Electronics and Communication Engineering for his esteemed supervision and guidance during the tenure of my internship work.

Also, I would like to express my gratitude and respect to our faculty advisor **Dr**. **Ajay Sir** and **Dr. Prashant Kumar Sir** and HOD of our Department **Dr. (Prof) Amit Prakash Sir** for their support, feedback and guidance throughout my intern course duration.

By

Kunal Mehta

**CONTENTS**

Title Page i

Candidate’s Declaration ii

Certificate iii

Acknowledgement iv

Contents v

Abstract vi

1. Introduction 1

2. Motivation 2

3. Theoretical Study 3

4. Algorithm 4

5. Experimental Study 5

6. Results and Discussions 6

7. Conclusion 10

8. References 11

**Design of Encoding – Decoding Algorithm and Implementation using Web Development and Data Structures**

**Abstract**

In this Project, the idea was to develop an algorithm that compresses input data to give a smaller output as the encoder and decoder are used in many electronics projects to compress the multiple numbers of inputs into a smaller number of outputs.

I have developed an experimental algorithm in inspiration with Huffman Coding technique to encode and decode data, and have made a web application to display the graph each time when data is inserted in accordance with the algorithm. The algorithm uses data structures such as max heap (priority queue), trees, arrays and web scripting languages such as JavaScript for its implementation.

* **Keywords**: Encoder, Decoder, Huffman Coding Algorithm, Graphs, Priority Queues, Trees.
* **Technology Used**: JavaScript , HTML , CSS , GITHUB Pages .
* **Source Link:** [**https://github.com/Kunalmehta99/industrial\_training\_nitjsr**](https://github.com/Kunalmehta99/industrial_training_nitjsr)
* **Live Demo:** [**https://kunalmehta99.github.io/industrial\_training\_nitjsr/**](https://kunalmehta99.github.io/industrial_training_nitjsr/)

**1. Introduction**

Now, in a world of having huge data, we use certain techniques to compress the data to store and transmit it. This saves a lot of space, time and money. One of the most common technique that is used in common world is Huffman Coding technique.

Huffman coding is a method of data compression that is independent of the data type, that is, the data could represent an image, audio or spreadsheet. This compression scheme is used in JPEG and MPEG-2. Huffman coding works by looking at the data stream that makes up the file to be compressed.

How is Huffman coding used to compress data?

Huffman coding is a form of lossless compression which makes files smaller using the frequency with which characters appear in a message. This works particularly well when characters appear multiple times in a string as these can then be represented using fewer bits. This reduces the overall size of a file as the most frequent character gets the smallest code and the least frequent character gets the largest code.

Keeping Huffman coding algorithm in mind, I am deducing an algorithm which encodes and decodes stream of characters in such a way that the most frequent character gets the smallest code and the least frequent character gets the largest code.

**2. Motivation**

In present days, whenever we have to decrease the size of files we use encoding-decoding technique.

Huffman coding is a very popular technique which makes files smaller using the frequency with which characters appear in a message.

My motive was to devise an algorithm to encode and decode the stream of characters

by using data structures and web development technologies to have working model of that algorithm.

My algorithm works on the fact that we have to assign the minimum bit to the character having the maximum frequency and keep on developing a tree by assigning bits on subsequent decreasing sequence according to the frequency count and finally encode the message in the form of bits.

**4. THEORATICAL STUDY**

**Design of Algorithm:**

* In Huffman Encoding algorithm we use a min heap i.e. a priority queue in which the value of frequency field is used to compare two nodes in min heap. Initially, the least frequent character is at root.
* Then we extract two nodes with the minimum frequency from the min heap and after that a new internal node with a frequency equal to the sum of the two nodes frequencies is created and we assign the first extracted node as its left child and the other extracted node as its right child and this node is added to the min heap.
* But in Experimental algorithm we are using a max heap that gives the value with the greater frequency and extract one node with the maximum frequency from the max heap each time to assign codes.
* And new internal node with a frequency equal to the sum of the all leftover nodes frequencies is created . The first extracted node (having maximum frequency) as its left child and the other extracted node as its right child. Add this node to the max heap.

**Algorithm for Experimental Encoding:**

* Create a leaf node for each unique character and build a min heap of all leaf nodes (Max Heap is used as a priority queue. The value of frequency field is used to compare two nodes in max heap. Initially, the most frequent character is at root)
* Extract one node with the maximum frequency from the max heap.
* Create a new internal node with a frequency equal to the sum of the all leftover nodes frequencies. Make the first extracted node as its left child and the other extracted node as its right child. Add this node to the max heap.
* Repeat steps#2 and #3 until the heap contains only one node. The remaining node is the last node and the tree is complete.

**Algorithm for Decoding:**

* To decode the encoded data, we require the Encoding tree. We iterate through the binary encoded data. To find character corresponding to current bits, we use following simple steps.
* We start from root and do following until a leaf is found.
* If current bit is 0, we move to left node of the tree.
* If the bit is 1, we move to right node of the tree.
* If during traversal, we encounter a leaf node, we print character of that particular leaf node and then again continue the iteration of the encoded data starting from step 1.

**5. Experimental Study**

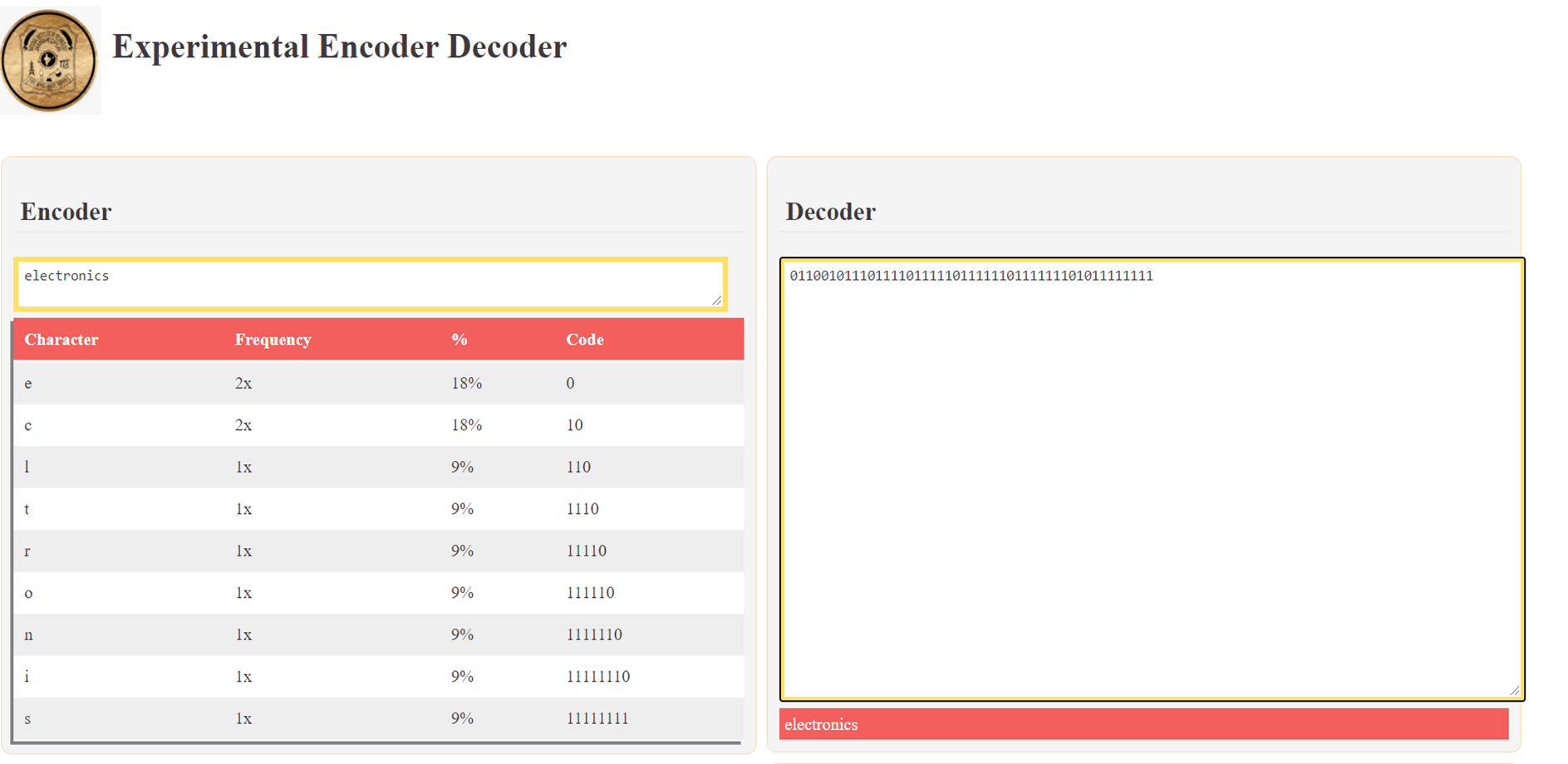
****

Fig.1. Encoded and Decoded message of “electronics”

The above figure 3.1 is obtained by encoding the stream of characters “electronics” from our experimental encoding algorithm. Frequency count and the percentage of occurrence of characters in the stream.

**6. Results And Discussion**

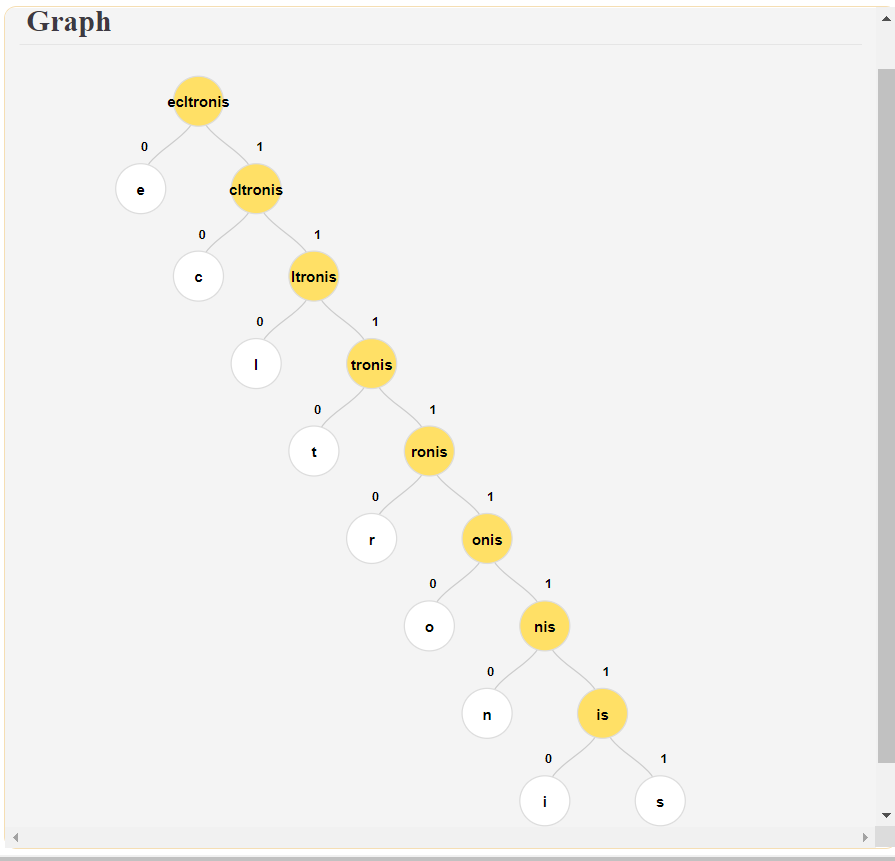


Fig 2. Graphical representation of the tree formed during encoding

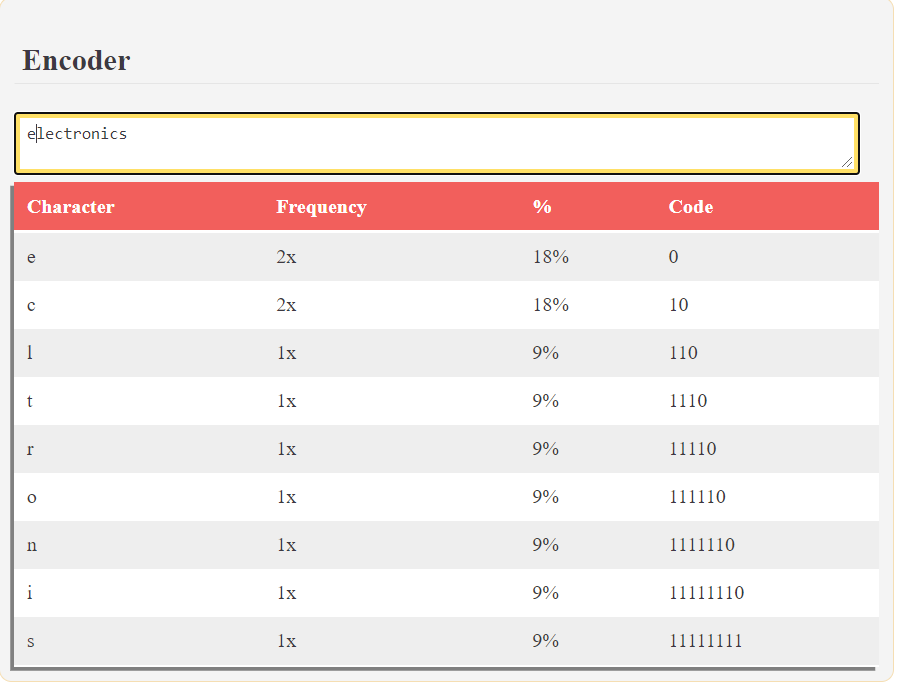
From the fig 4.1.1, already write operation has been done and we are getting the encoded result in the form of stream of bits. We also have the tree and the assigned bits on the screen,

**Comparing Input file size and Output file size:**

* Comparing the input file size and the Experimental encoded output file. We can calculate the size of the output data in a simple way. Lets say our input is a string “electronics” and is stored in a file input.txt.

Input File Size:

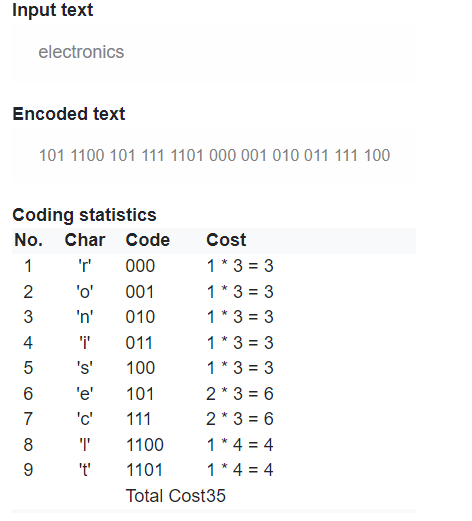
* Input: “electronics"
* Total number of characters i.e. input length: 13
* Size: 11 characters occurrences \* 8 bits = 88 bits or 11 bytes.



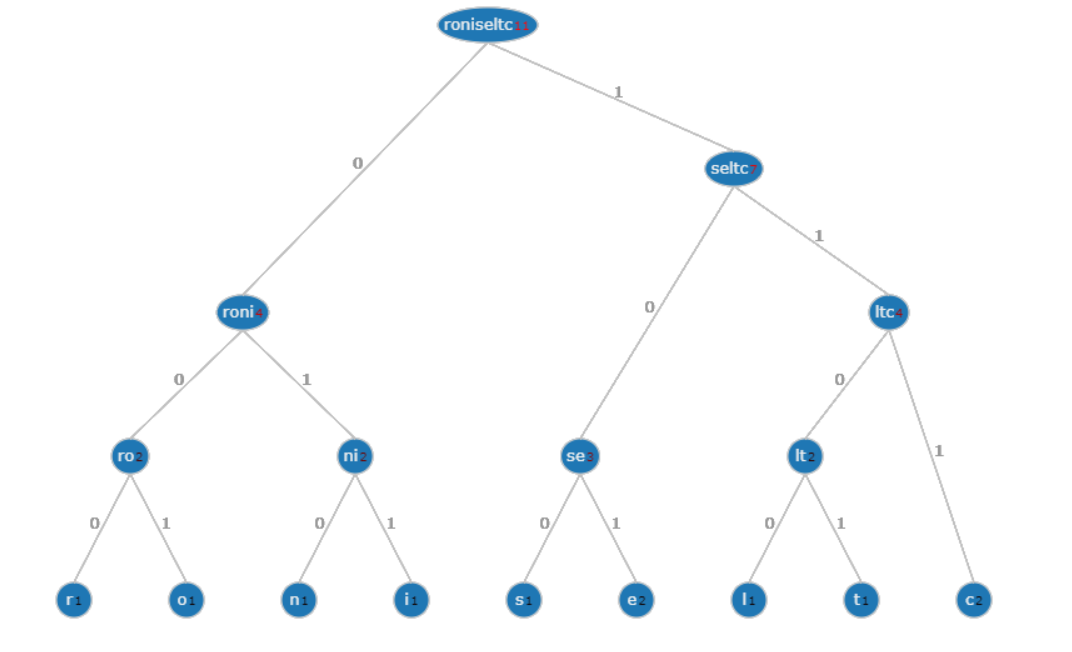
Output File Size:

* Total Sum: 2x1 + 2x2 + (3 + 4 +5 + 6 + 7 + 8 + 8) x 1 = 47 bits or 6 bytes
* Compression ratio: (88 – 47) / 88 = 0.47
* Compression Percentage: 47%

**Comparing Experimental algorithm with Huffman coding algorithm :**



* Total Sum for Huffman Coding Algorithm = 35 bits or 5 bytes
* Compression ratio: 0.602
* Compression Percentage: 60.2 %
* Therefore, our experimental algorithm is not efficient in comparison to Huffman algorithm

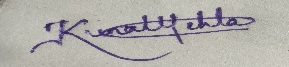
**Graphical Representtion:**

**7. CONCLUSION**

The comparative results for the encoded message and the normal message are obtained and our experimental algorithm is successfully able to compress the stream of characters and thus we are able to save the memory by 47% for the test case of stream of characters. But when it is compared to Huffman coding , Huffman outshines the experimental algorithm and thus, this also proves that Huffman gives the most compressed results.

**8. REFERENCE**

1. Huffman Coding Theory - <https://en.wikipedia.org/wiki/Huffman_coding>
2. Visualization of Huffman Coding – <https://demo.tinyray.com/huffman>
3. Algorithmic codes : <https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/>



Signature of Students