University of Missouri-Kansas City

CS 303 Data Structures

Section 2

Project 2

Morse Code

by

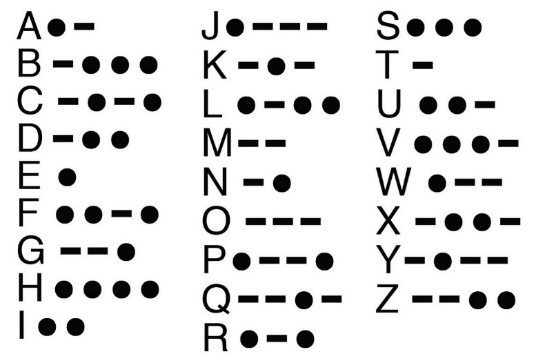
Tuyen Vu

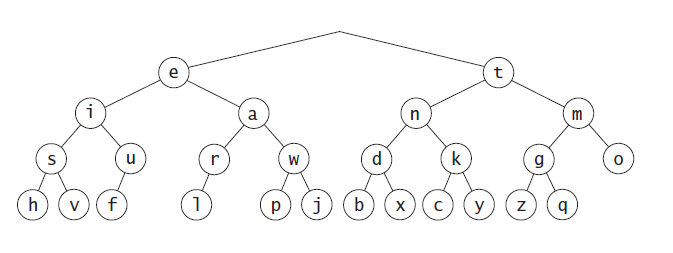
1. **Acknowledgment**

Morse code is a common code that is used to encode messages consisting of letters and digits. Each letter consists of a series of dots and dashes; for example, the code for the letter a is • – and the code for the letter b is – •••. Store each letter of the alphabet in a node of a binary tree of depth 4. The root node is at depth 0 and stores no letter. The left node at depth 1 stores the letter e (code is •) and the right node stores the letter t (code is –).

In this project, I am going to build a Morse Tree based on provided Morse code from the table below. The left edges represent dots, and the right edges represent dashes. Each node is a letter. After building the Morse Tree, I am going to use it to encode and decode a message.

Here is how the Morse Tree look like:





1. **Time Complexity**

Building the Morse Tree takes O(nlogn). Printing out the tree and finding a letter is O(logn). Encoding and decoding process have the same time complexity O(nlogn).

Therefore, the overall time complexity is O(nlogn).

1. **UML Diagram**

|  |
| --- |
| Morse\_Tree |
| -root: Node\* |
| **+**Morse\_Tree()  +get\_root(): Node\*  +full\_Morse(temp:Node\*, strTemp:string): string  +find(rootTemp:Node\*, strTemp:string, letter:char): string  +letters\_to\_morse(letters:string): string  +morse\_to\_letters(morseCode:string):string |

|  |
| --- |
| <<struct>>  Node |
| +data:char  +nextLeft: Node\*  +nextRight: Node\*  +back: Node\* |
| **+**Node() |

1. **Output**

