CONTENTS

ACKNOWLEDGEMENTS i
LIST OF TABLES iii
LIST OF FIGURES vi
LIST OF ABBREVIATIONS xvi

ABSTRACT xix

CHAPTERS

I INTRODUCTION

1.1 Introduction 1
1.2 Objective of Thesis 2
1.3 Information Theory and Background 3
  1.3.1 History of Data Compression 3
1.4 Data Compression 4
1.5 Classification of Data Compression Algorithms 5
  1.5.1 Lossless Data Compression Algorithm 5
    1.5.1.1 Dictionary Based Compression Algorithm 7
    1.5.1.2 Statistical Based Data Compression Algorithm 8
    1.5.1.3 Transformation Based Data Compression Algorithm 9
  1.5.2 Lossy Data Compression Algorithm 9
1.6 Parameters to Measure the Performance of Compression Algorithms 10
1.7 Big-O Notation 13
  1.7.1 Types of Order 14
    1.7.2 Computational Complexity and Logarithms 15
1.8 Bench Mark Files 16
1.9 Time complexity Evaluator Tool 19
1.10 Experimental Setup 20
1.11 Frame work of This Thesis 20

II LITERATURE REVIEW

2.1 Background of the Study 23
  2.1.1 Grammar Based LZ Optimization 23
### III Survey of Existing Lossless Algorithms

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>33</td>
</tr>
<tr>
<td>3.2</td>
<td>The Huffman Coding Algorithm</td>
<td>33</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Performance Analysis of Huffman Coding</td>
<td>36</td>
</tr>
<tr>
<td>3.3</td>
<td>Run Length Encoding Algorithm (RLE)</td>
<td>40</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Performance Analysis of Run Length Encoding Algorithm</td>
<td>43</td>
</tr>
<tr>
<td>3.4</td>
<td>Arithmetic Coding Algorithm</td>
<td>47</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Performance Analysis of Arithmetic Coding</td>
<td>50</td>
</tr>
<tr>
<td>3.5</td>
<td>Lempel Ziv 77 Algorithm</td>
<td>55</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Performance Analysis of Lempel Ziv 77</td>
<td>57</td>
</tr>
<tr>
<td>3.6</td>
<td>Lempel Ziv Welch Algorithm</td>
<td>62</td>
</tr>
<tr>
<td>3.7</td>
<td>Quick sort</td>
<td>68</td>
</tr>
<tr>
<td>3.8</td>
<td>Experimental Results</td>
<td>70</td>
</tr>
<tr>
<td>3.9</td>
<td>Summary</td>
<td>74</td>
</tr>
</tbody>
</table>

### IV A Novel Sorting Algorithm: Binary Insertion Sort

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>75</td>
</tr>
<tr>
<td>4.2</td>
<td>Binary Insertion Sort</td>
<td>77</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Binary Insertion Sort Average Case Analysis</td>
<td>84</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Binary Insertion Sort Worst Case Analysis</td>
<td>85</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Binary Insertion Sort Best Case Analysis</td>
<td>86</td>
</tr>
</tbody>
</table>
4.3 Experimentations and results

4.4 Summary

V DATA STRUCTURES AND LZW COMPRESSION ALGORITHM

5.1 Introduction

5.2 Linear array and Linear Search

5.2.1 Linear Search in an Unsorted Array

5.2.2 Computational Complexity Analysis of Linear Search in an Unsorted Array

5.2.2.1 Time complexity Analysis for a Successful Search

5.2.2.2 Time complexity Analysis of an unsuccessful Search

5.2.3 LZW Compression linear array Implementation with linear search

5.2.3.1 Time complexity Analysis of LZW linear array Implementations

5.2.4 LZW Decompression Linear Array Implementation with Linear Search

5.2.4.1 Time complexity Analysis of LZW linear array Implementations

5.3 Binary Search Trees (BST)

5.3.1 Computational Complexity Analysis of BST

5.3.1.1 Time complexity Analysis for a Successful Search

5.3.1.2 Time complexity Analysis of an unsuccessful Search

5.3.2 Binary search tree Implementation of LZW Encoding Algorithm

5.3.2.1 Time complexity Analysis of LZW BST Implementations

5.4 Chained Hash table

5.4.1 Hashing function

5.4.2 Time complexity evaluation of Chained Hash table

5.4.2.1 Time complexity Analysis for a Successful Search

5.4.2.2 Time complexity Analysis for an Unsuccessful Search

5.4.3 LZW Compression Chained Hash Table Implementation

5.4.3.1 Time complexity Analysis LZW Chained Hash table
MDLZP APPROACH ON LZW (MDLZW) COMPRESSION ALGORITHM AND ITS DATA STRUCTURE IMPLEMENTATION

6.1 Introduction
6.2 MDLZW Linear array Implementation
  6.2.1 Time complexity Analysis of MDLZW linear array Implementation
6.3 MDLZW Binary Search Tree (BST) Implementation
  6.3.1 Time complexity Analysis of MDLZW with Binary search Tree Implementation
6.4 MDLZW Chained Hash Table Implementation
  6.4.1 Time complexity Analysis of MDLZW with Chained Hash table Implementation
6.5 MDLZW Binary Insertion Sort (BIS) Implementation
  6.5.1 Time complexity Analysis of MDLZW with BIS Implementation
6.6 Experimentations and results
6.7 Summary

INDEXED K TWIN NEIGHBOUR CLUSTERING ALGORITHM

7.1 Introduction
7.2 Clustering Techniques
  7.2.1 Clustering Paradigms
    7.2.1.1 Partitioning Algorithm
7.3 Indexed K Nearest Twin Neighbour Clustering Algorithm (IKNTN)
  7.3.1 Computational Complexity Analysis of IKNTN Algorithm
    7.3.1.1 Linear array Implementation of IKNTN Algorithm
    7.3.1.2 BST Implementation of IKNTN Algorithm
    7.3.1.3 Hash table Implementation of IKNTN Algorithm
    7.3.1.4 BIS Implementation of IKNTN Algorithm
7.3.2 Space complexity of IKNTN Algorithm 162

7.4 Experimental Results 162

7.5 Summary 167

VIII  **LZW with Indexed K Nearest Twin Neighbour (IKNTN) algorithm**

8.1 Introduction 168

8.2 LZW with Indexed K Nearest Twin Neighbour (IKNTN) algorithm 169

8.3 IKNTN_LZW with various implementations 172

8.4 Computational Complexity Analysis of Various Implementations 182

8.5 Summary 187

IX  **Enhancement of MDLZW with Indexed K Nearest Twin Neighbour Algorithm**

9.1 Introduction 188

9.2 MDLZW Dictionaries Clustering using IKNTN 189

9.2.1 Computational Complexity Analysis of MDLZW_IKNTN 192

9.2.1.1 Computational Complexity of MDLZW_IKNTN with Linear Array Implementations 192

9.2.1.2 Computational Complexity of MDLZW_IKNTN with BST implementations 196

9.2.1.3 Computational Complexity of MDLZW_IKNTN with Chained Hash Table implementations 200

9.2.1.4 Computational Complexity of MDLZW_IKNTN with BIS implementations 203

9.3 LZW Clustered Dictionary Into Multiple Dictionary(MD) 207

9.3.1 Computational Complexity analysis of IKNTN_MDLZW 209

9.3.1.1 Computational Complexity Linear array Implementations 209

9.3.1.2 Computational Complexity BST Implementations 214

9.3.1.3 Computational Complexity Chained Hash Table Implementations 217

9.3.1.4 Computational BIS Implementations 220

9.4 Experimental Results 223

9.5 Summary 227
# RESULTS AND DISCUSSION

10.1 Introduction 228
10.2 Results 229

10.2.1 Comparative Analysis of Lossless Data Compression Algorithm 229
10.2.2 Comparative Analysis of Data structures Implementation of LZW 232
10.2.3 Comparative Analysis of Data structures Implementation of MDLZW 233
10.2.4 Computational Complexity Analysis of Data Structure After Clustering with IKNTN 234
10.2.5 Computational Complexity analysis of LZW with IKNTN 236
10.2.6 Computational Complexity Analysis of MDLZW with IKNTN 237

10.4 Summary 242

# CONCLUSION AND FUTURE ENHANCEMENT

11.1 Conclusion 243
11.2 Limitations 244
11.3 Future Enhancements 244

# REFERENCES

# LIST OF PUBLICATIONS

# APPENDICES

A. Prototype for IKNTN Algorithm with Linear Array Implementation XV

B. Prototype for IKNTN Algorithm with Binary Search Implementation XXV