# The Implementation and Comparison of the BCCBT Data Compression Algorithm

Aiden Taylor - B.Sc. in Computer Science
Noah Pinel - B.Sc. in Computer Science
Ty Irving - B.Sc. in Computer Science

Apr. 11th, 2023

1. What specific type of Binary Tree is used in the implementation of the BCCBT Data Compression Algorithm? Describe at least one discussed property of this type of Binary Tree.
2. Given the specific binary tree needed for the BCCBT algorithm, where the tree's nodes correspond to symbols in an arbitrary alphabet $\Sigma$. Denote a symbol $\phi$, such that $\phi$ is in our arbitrary alphabet $\Sigma$. Note, $\phi$ is **NOT** the root of the tree. How is the bit code generated for the symbol $\phi$?
3. Does the BCCBT Data Compression Algorithm make use of the frequency/probability of each unique symbol in the source file? Explain why or why not.

# Bit Code Complete Binary Tree (BCCBT)

## PSEUDOCODE OF THE BCCBT ALGORITHM

**ENCODING**

```
Get the frequency of each symbol from the input stream
Set the frequency table to the frequency of each symbol
Create a complete binary tree using the frequency table
Set the bit codes according to where the symbols are in the tree
While more symbols to read from the input stream
      Read one symbol from the input stream
      Write the symbol's bit code to the bit code stream
      Write the length of the bit code to the level stream
Compress the level stream with a lossless algorithm
Write the frequency table to the output stream
Write the compressed level stream and the bit code stream to the output
stream
```

**DECODING**

```
Read the frequency table from the input stream
Create a complete binary tree using the frequency table
Read the compressed level stream from the input stream
Uncompress the compressed level stream
Read the bit code stream from the input stream
While more levels to read from the level stream
      Read one level from the level stream
      Read level bits from bit code stream
      Find the symbol in the complete binary tree using the level and
the bit code
      Write the symbol to the output stream
```

**Table 3-6**

| Symbol | Frequency |
|---|---|
| *a* | 32 |
| *b* | 55 |
| *c* | 4 |
| *d* | 19 |
| *e* | 37 |
| *f* | 26 |
| *g* | 9 |
| *h* | 7 |

If we now were to create a complete binary tree of Table 3-6, we would get the following tree:

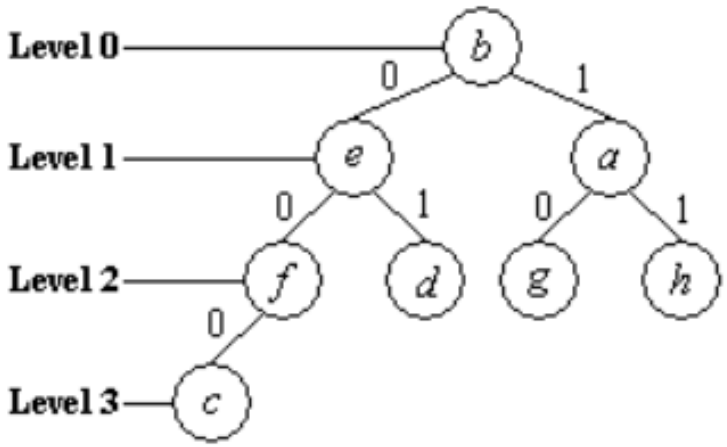



**Figure 3-5** Complete binary tree

Properties of Complete Binary Trees:

- All levels are completely full except possibly the lowest level.
- Filled from left to right at each level. i.e. Tree leans left.
- The number of nodes at level $n$ is $2^n$.

**Table 3-6**

| Symbol | Frequency |
|---|---|
| *a* | 32 |
| *b* | 55 |
| *c* | 4 |
| *d* | 19 |
| *e* | 37 |
| *f* | 26 |
| *g* | 9 |
| *h* | 7 |

If we now were to create a complete binary tree of Table 3-6, we would get the following tree:

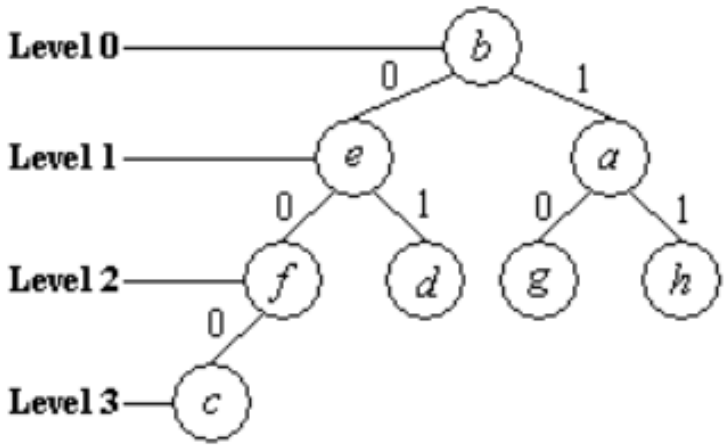
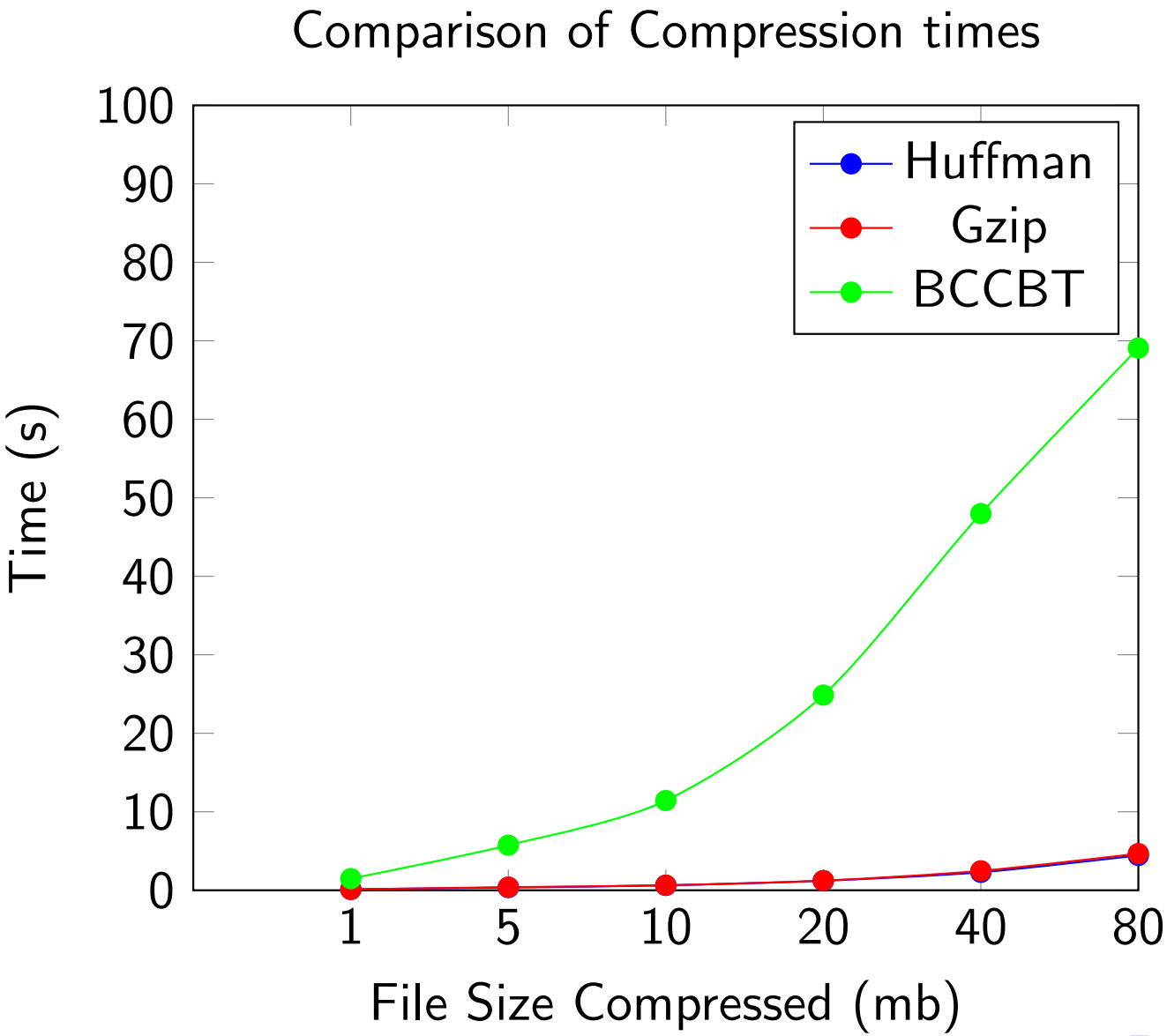



**Figure 3-5** Complete binary tree

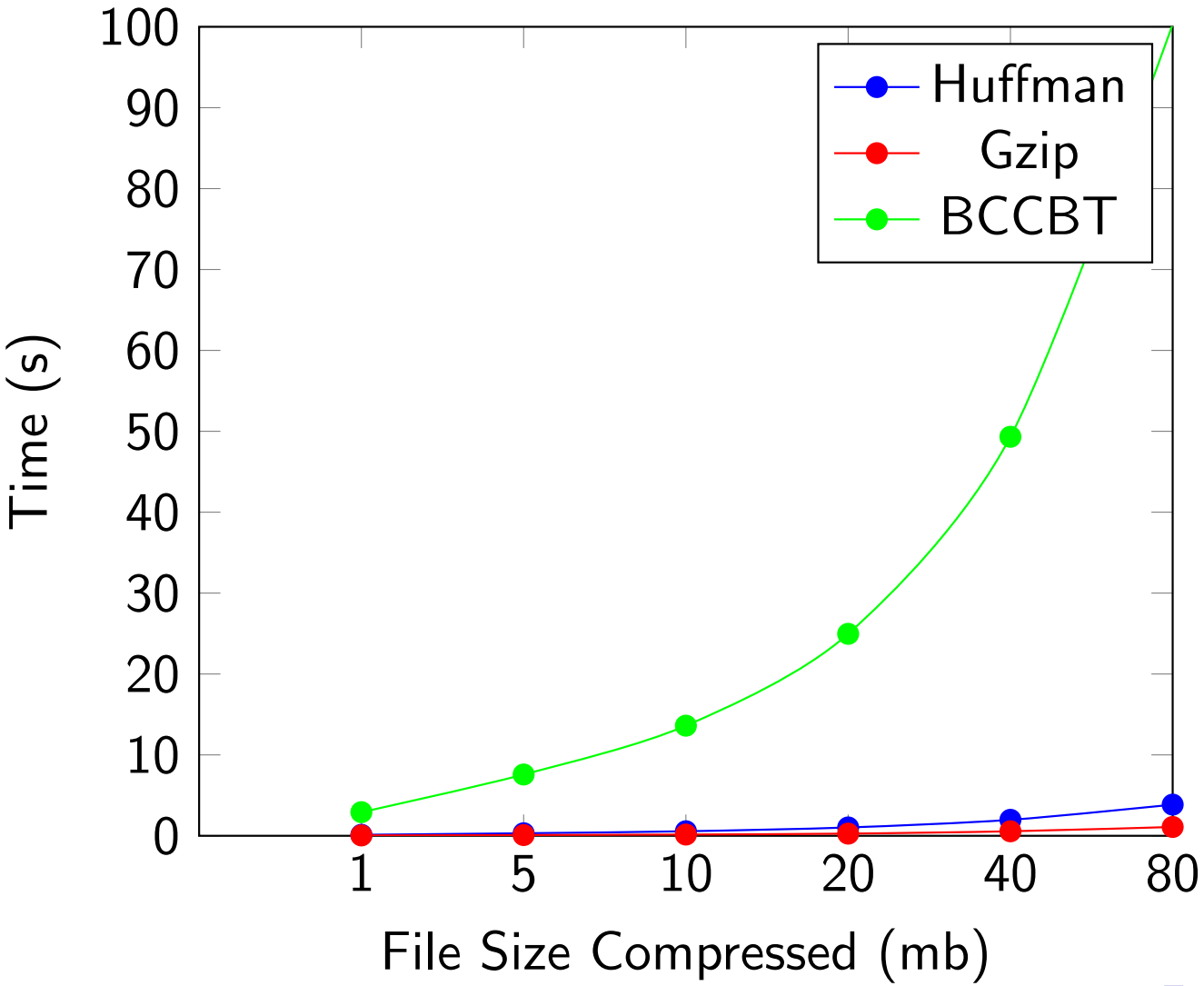
Noahs Part

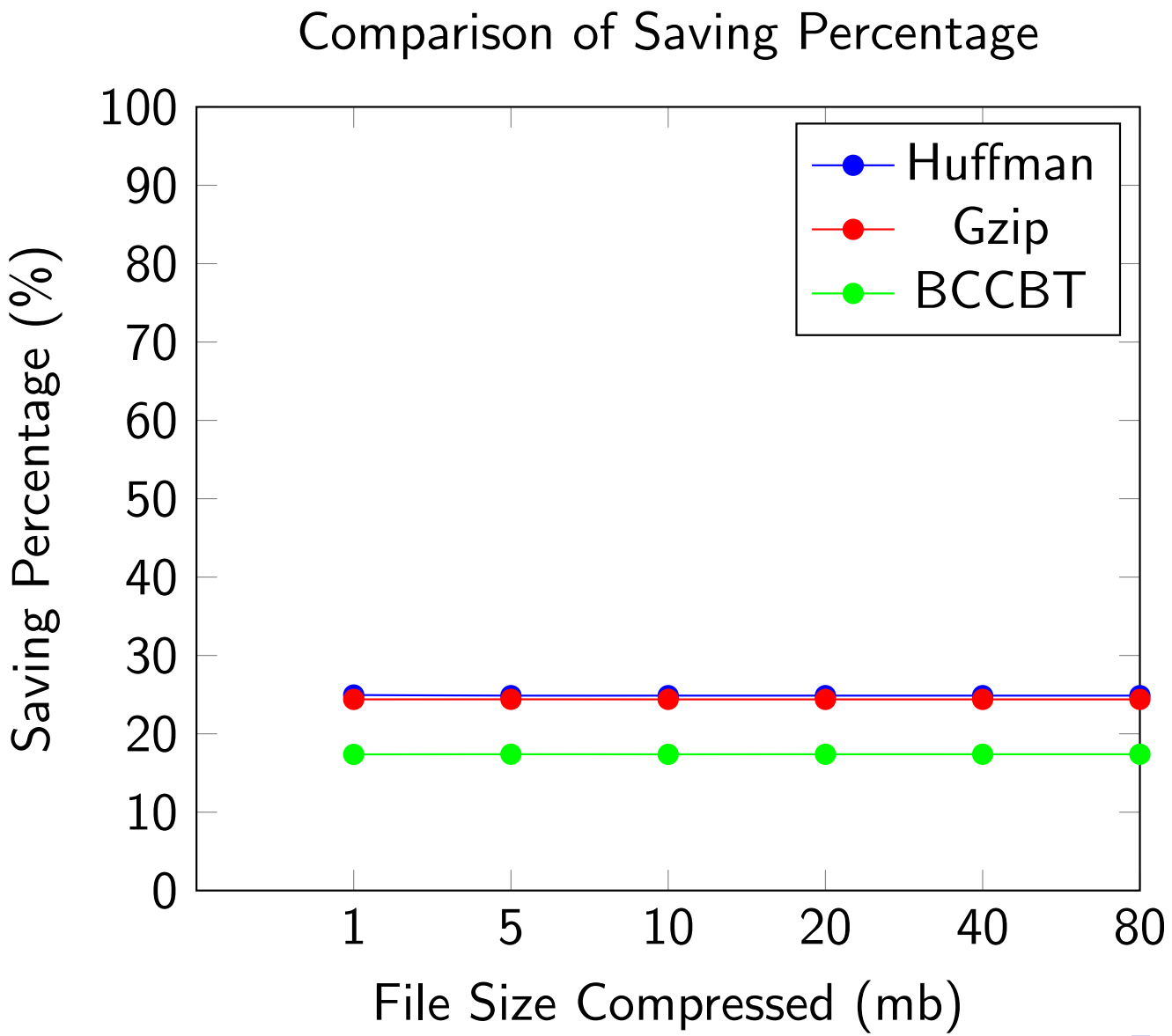
Some of the factors we will be using to analyze and compare these algorithms are as follows:

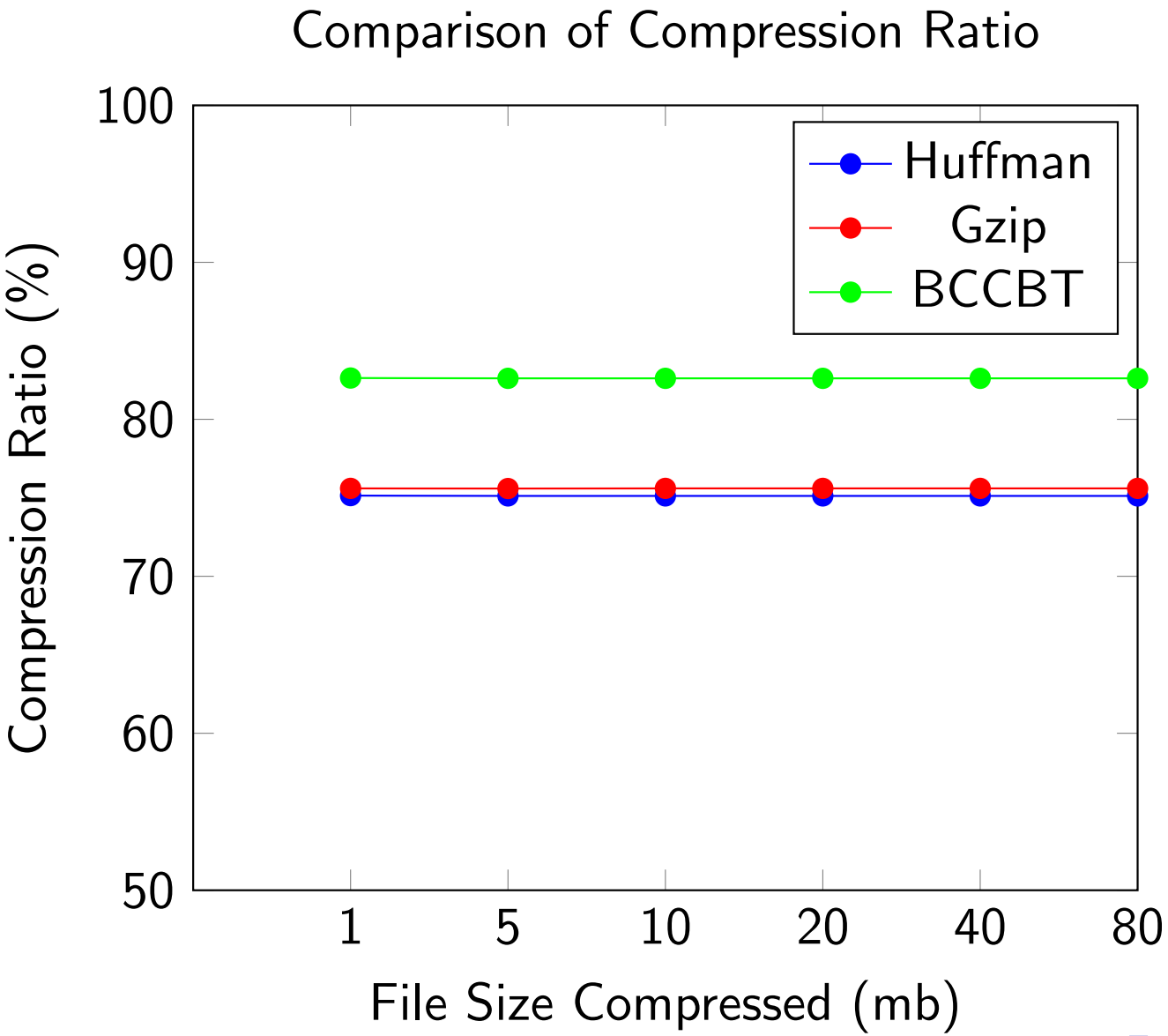
1. Compression Time
2. Decompression Time
3. Saving Percentage $= \frac{Original\ \ File\ \ Size\ -\ \ Compressed\ \ File\ \ Size}{Original\ \ File\ \ Size}$
4. Compression Ratio $= \frac{Compressed\ \ File\ \ Size}{Original\ \ File\ \ Size}$


Comparison of Compression times
Time (s)
100
90
80
70
60
50
40
30
20
10
0
1
5
10
20
40
80
File Size Compressed (mb)
Huffman
Gzip
BCCBT

Comparison of Decompression times




Comparison of Saving Percentage
Saving Percentage (%)
100
90
80
70
60
50
40
30
20
10
0
1
5
10
20
40
80
File Size Compressed (mb)
Huffman
Gzip
BCCBT


Comparison of Compression Ratio
Compression Ratio (%)
100
90
80
70
60
50
1
5
10
20
40
80
File Size Compressed (mb)
Huffman
Gzip
BCCBT

Thank you for listening! Any questions?